



DUDLEY KNOX LIBRARY
NATIONAL GRADUATE SCHOOL
MOSCOW, CALIFORNIA 93943

NAVAL POSTGRADUATE SCHOOL

Monterey, California



THESIS

INTERACTIVE IMPLEMENTATION OF THE
OPTIMAL SYSTEMS CONTROL DESIGN PROGRAM (OPTSYSX)
ON THE IBM 3033

by

John Gustav Hoden II

March 1984

Thesis Advisor:

Daniel J. Collins

Approved for public release; distribution unlimited.

1215202

REPORT DOCUMENTATION PAGE

READ INSTRUCTIONS
BEFORE COMPLETING FORM

1. REPORT NUMBER	2. GOVT ACCESSION NO.	3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle) INTERACTIVE IMPLEMENTATION OF THE OPTIMAL SYSTEMS CONTROL DESIGN PROGRAM (OPTSYSX) ON THE IBM 3033		5. TYPE OF REPORT & PERIOD COVERED Master's Thesis March 1984
7. AUTHOR(s) John G. Hoden II		6. PERFORMING ORG. REPORT NUMBER
9. PERFORMING ORGANIZATION NAME AND ADDRESS Naval Postgraduate School Monterey, California 93943		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS
11. CONTROLLING OFFICE NAME AND ADDRESS Naval Postgraduate School Monterey, California 93943		12. REPORT DATE March 1984
14. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office)		13. NUMBER OF PAGES 145
16. DISTRIBUTION STATEMENT (of this Report)		15. SECURITY CLASS. (of this report) Unclassified
17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)		15a. DECLASSIFICATION/DOWNGRADING SCHEDULE
18. SUPPLEMENTARY NOTES		
19. KEY WORDS (Continue on reverse side if necessary and identify by block number) Optimal Systems Control Systems Control Control Systems		
20. ABSTRACT (Continue on reverse side if necessary and identify by block number) This thesis discusses the modification of an existing Optimal Systems Control FORTRAN program (OPTSYS) originally obtained from Professor Arthur E. Bryson of Stanford University, The modified FORTRAN program (OPTSYSX) is now designed to run completely interactively under VM/CMS on the IBM 3033 and is considered completely compatible with similar operating systems.		

Program capabilities include: complete eigensystem analysis; the ability to perform computations on very large multi-variable systems; controller, filter, regulator and compensator synthesis; transfer function analysis; steady-state gain determinations; and modal analysis.

The program permits users to rapidly carry out simulation, analysis, and design of all classes of optimal systems control problems in a totally interactive mode. Examples of various types of problems are worked out during individual terminal sessions.

Approved for public release; distribution unlimited.

Interactive Implementation of the
Optimal Systems Control Program (OPTSYSX)
on the IBM/3033

by

John G. Hoden
Lieutenant Commander, United States Navy
B.A., University of Minnesota, Duluth, 1970

Submitted in partial fulfillment of the
requirements for the degree of

MASTER OF SCIENCE IN AERONAUTICAL ENGINEERING

from the

NAVAL POSTGRADUATE SCHOOL
March 1984

AESTHACT

This thesis discusses the modification of an existing Optimal Systems Control FORTRAN Program (OPTSYS) originally obtained from Professor Arthur E. Bryson of Stanford University.

The modified FORTRAN program (OPTSYSX) is now designed to run completely interactively under VM/CMS on the IBM 3033 and is considered completely compatible with similar operating systems.

Program capabilities include: complete eigensystem analysis; the ability to perform computations on very large multivariate systems; controller, filter, regulator and compensator synthesis; transfer function analysis; steady-state gain determination; and nodal analysis.

The program permits users to rapidly carry out simulation, analysis, and design of all classes of Optimal Systems Control problems in a totally interactive mode. Examples of various types of problems are worked out during individual terminal sessions.

TABLE OF CONTENTS

I.	INTRODUCTION	9
II.	THE OPTSYSX COMPUTER PROGRAM	11
A.	INTRODUCTION	11
B.	SYSTEM/MODEL DESCRIPTION	11
C.	PROGRAM OUTPUT	13
1.	Open-Loop Eigensystem Calculations	13
2.	Regulator Synthesis Calculations	13
3.	Filter Synthesis Calculations	14
4.	Stationary Closed-Loop Calculations	15
D.	SOLUTION ALGORITHM	15
E.	PROGRAM OVERVIEW	16
1.	Problem Description/Program Flow Control	17
2.	Interactive Data Input	17
3.	Calculation Sequencing	17
4.	QR Algorithm Transformation	17
5.	Riccati Equation Calculations	17
6.	Model Calculations	18
7.	Transfer Function Calculations	18
8.	Data Output	18
III.	INTERACTIVE PROGRAM OPERATION	19
A.	DESIGN CONSIDERATIONS	19
B.	PROGRAM LANGUAGE	19
C.	GENERAL PROGRAM MODIFICATIONS	20
D.	INTERACTIVE PROGRAM DEVELOPMENT	21
1.	Program Flow Control	21
2.	General Input Sequencing Requirements	22

3. Interactive Data Input	23
4. Saving Interactive Input	24
5. User-Defined Input Files	24
6. Internal Data Generation	25
7. Data Entry Correction	26
E. USER-ERROR PROTECTION FEATURES	27
1. Data Type Conversion Errors	27
2. Inconsistent Program Control Flag Errors	28
IV. PROGRAM USE AND EXAMPLES	29
A. CPEN-100P EIGENSYSTEM ANALYSIS	29
B. REGULATOR SYNTHESIS	33
C. FILTER SYNTHESIS	43
D. EXAMPLE OF PROGRAM FAILURE	53
V. CONCLUSIONS AND RECOMMENDATIONS	69
A. CONCLUSIONS	69
B. RECOMMENDATIONS	70
1. Program Availability	70
2. Computer Graphics	70
3. Further Modifications	70
4. Program Application	71
LIST OF REFERENCES	143
BIBLIOGRAPHY	144
INITIAL DISTRIBUTION LIST	145

SYMBOLS

A = state (N_s, N_s) or output (N_o, N_o) weighting matrix
B = control (N_c, N_c) weighting matrix
C = control gain matrix (N_c, N_s)
D = control (N_c, N_c) or noise (N_o, N_g) feedforward
matrix
E = expected value
F = open-loop dynamics matrix (N_s, N_s)
G = control distribution matrix (N_s, N_c)
GAM = state disturbance distribution matrix (N_s, N_g)
H = measurement scaling matrix (N_o, N_s)
K = estimator gain matrix (N_s, N_{ob})
Nc = number of controls
Ng = number of process noise sources
Ns = number of states
No = number of observations or measurements
P = covariance matrix of estimate error (N_s, N_s)
Q = white process noise covariance matrix (N_g, N_g)
R = white meas. noise covariance matrix (N_o, N_o)
S = steady-state covariance matrix of control (N_c, N_c)
u = control vector ($N_c, 1$)
v = white measurement noise vector ($N_o, 1$), with
zero mean and covariance matrix R
w = white process noise vector ($N_g, 1$), with
zero mean and covariance matrix Q
w0 = constant disturbance vector ($N_g, 1$)
x = state vector ($N_s, 1$)
xe = estimate of state vector ($N_s, 1$)
y = output/measurement vector ($N_o, 1$)

ACKNOWLEDGEMENT

I would like to express my sincere appreciation to Professor L.J. Collins, whose assistance and encouragement contributed immeasurably to this research.

I wish to dedicate this thesis to my wife, Brenda. Without her constant love, support, and understanding this work would not have been possible.

I. INTRODUCTION

The purpose of this thesis is to describe the extensive modification and improvement of an existing FORTRAN program (OPTSYS) used in the study, design, and application of Optimal Systems Control theory.

This optimal systems control program was originally developed by Hall [Ref. 1] in 1971 to support his research in rotary-wing VTOL aircraft control systems. The latest program modifications were made by Walker [Ref. 2] and Liu [Ref. 3] of Stanford University, to OPTSYS 4 and CPTSYS 5 respectively. These program versions performed quite satisfactorily in the batch environment, but exhibited varying degrees of user hostility due to data input format requirements and incomplete program documentation.

The original intent of this work was to adapt Walker's modified version of CPTSYS to run interactively under VM/CMS on the IBM 3033; however, the extensive modifications of OPTSYSX now represent a high-level interactive applications software system capable of integrated simulation, analysis, synthesis and design of broad classes of optimal systems control problems. With OPTSYSX users may now evaluate various specialized optimal systems control applications, relieved of the burden of lengthy mathematical program development.

It is assumed that the reader/user is familiar with the basic concepts of Control Theory and Optimal Systems Design. The problem descriptions and program development follow the terminology and symbol/naming conventions of Bryson [Ref. 4].

An explanation of the basic system of equations, the terms and symbolism used, and a program overview including the general methods of solution are presented first.

Interactive program development is then discussed, with an explanation of several alternate options available for data input.

This work concludes with examples of various types of problems demonstrated in the interactive mode, including a copy of each terminal session with the final results. A complete program listing is included in Appendix A.

II. THE OPTSYSX COMPUTER PROGRAM

A. INTRODUCTION

OPTSYSX is a double-precision FORTRAN program employing modern control theory analysis techniques. Although the program was originally written to synthesize controllers for rotary-wing VTOL aircraft [Ref. 5], it has been extensively modified to enable controller, filter, and regulator synthesis as well as transfer function and modal analysis on other types of large, multi-variable systems of equations. The program modifications described in this work now allow rapid numerical computer analysis in a completely interactive mode.

E. SYSTEM/MODEL DESCRIPTION

OPTSYSX treats a linear stationary system model:

$$\dot{x} = [F]x + [G]u + [\text{GAMMA}]w \quad (2.1)$$

output equation

$$y = [H]x + [D]u \quad (2.2)$$

measurement equation

$$z = [H]x + [D]w + v \quad (2.3)$$

where

u = control vector ($m \times 1$)

v = white measurement noise vector ($p \times 1$)

w = white process noise vector ($q \times 1$)

x = state vector ($n \times 1$)


```

y = output vector (p X 1)
z = measurement vector (q X 1)

```

$[F]$ is the open-loop dynamics matrix (system matrix or plant matrix); $[G]$ is the control distribution matrix; $[\Gamma]$ is the process noise distribution matrix; $[H]$ is the measurement distribution matrix; and $[D]$ may represent a feed-forward distribution matrix of either the process noise vector (w), or the control vector (u).

The w vector has zero mean value, and a covariance matrix $[Q]$, where:

$$E(w) = 0 \quad (2.4)$$

and

$$[Q] = E[ww^T] \quad (2.5)$$

The v vector has zero mean value and a covariance matrix $[R]$, where:

$$E(v) = 0 \quad (2.6)$$

and

$$[R] = E[vv^T] \quad (2.7)$$

The quadratic performance (or cost) index for the linear quadratic regulator is the expected value of:

$$J = 1/2 \int \{y^T [A]y + u^T [B]u\} dt \quad (2.8)$$

in the statistical steady-state, where $[A]$ represents an output cost matrix (a weighting on the output variables);

and $[B]$ is a control cost (or control weighting coefficient) matrix.

If full state weighting is desired, $[H]$ is represented by the identity matrix $[I]$.

C. EECGFAM OUTPUT

1. Open-Loop Eicensystem Calculations

The initial portion of OPTSYSX output includes the program flow control flags set by the user for that particular run, the system of equations being modeled, and the open-loop eigenvalue and eigenvector calculations of the $[F]$ matrix.

2. Regulator Synthesis Calculations

In the solution to the optimal regulator problem, full state variable feedback is assumed where:

$$[C] = [B^{-1}][G^T][S] \quad (2.9)$$

and

$$u = -[C]x \quad (2.10)$$

The control gain $[C]$ is a matrix of optimal gains which minimize the cost index expressed in equation (2.8).

For optimal regulator synthesis problems, program output includes the closed-loop eigenvalues and eigenvectors; the control gain $[C]$; the closed-loop dynamics matrix $[F-GC]$; and the steady-state gain matrix $[S]$, where $[S]$ is the steady-state solution to the algebraic Riccati equation:

$$S[F] + [F^T]S - S[G][B^{-1}][G^T]S + [H^T][A][H] = 0 \quad (2.11)$$

3. Filter Synthesis Calculations

A Kalman filter or Estimator which describes a continuous time system may be written as:

$$\dot{\hat{x}} = [F]\hat{x} + [K](z - [H]\hat{x}) \quad (2.12)$$

where:

\hat{x} is the state estimate

$[K]$ is a matrix of filter gains,

and the state covariance is described by:

$$E(xx^T) = E(\hat{x}\hat{x}^T) + [P] \quad (2.13)$$

The filter gain matrix $[K]$ of equation (2.12) is obtained from the relationship:

$$[K] = [P][H^T][R^{-1}] \quad (2.14)$$

where $[F]$ is the steady-state solution to the algebraic Riccati equation:

$$[F]F + P[F^T] - P[F^T][R^{-1}][H]F + [G][Q][G^T] = 0 \quad (2.15)$$

representing the error covariance of the estimate \hat{x} . The control covariance is the expected value described by:

$$E(uu^T) = [C]\hat{x}[C^T] \quad (2.16)$$

For the Kalman filter/optimal estimator synthesis problem, CFTSYSX output includes the eigenvalues and eigenvectors of the optimal estimator (Kalman filter); the filter gains $[K]$; the error covariance matrix $[P]$; the covariance

matrix of the estimate \hat{X} ; the state covariance matrix $[X]$ described in equation (2.13), where

$$[X] = E[XX^T] \quad (2.17)$$

and the control covariance matrix $[U]$ described in equation (2.16).

4. Stationary Closed-Loop Calculations

The stationary response of both state and control are presented as root-mean-square values of the state and control covariance matrices $[X]$ and $[U]$ described in equations (2.17) and (2.16) respectively.

E. SOLUTION ALGORITHM

One of the fundamental techniques necessary to quadratic synthesis of optimal control systems is the steady-state solution of the algebraic Riccati equation. This is a non-trivial task due to the iterative nature of the solution.

The steady-state solution by any quadrature method necessitates time increment selection no greater than some fraction of the shortest period of the closed-loop system; imposing a significant computer solution time expenditure on the user as well as the requiring an extensive amount of computer storage capability due to the matrix expansion factor involved. Further, the possible necessity of a time-varying solution of these equations for optimal open loop control or estimation requires the inversion of an $n \times n$ matrix for each time increment where an unsteady solution is desired.

A powerful and efficient alternate method of solution was developed by Bryson and Hall [Ref. 7], based on eigenvector decomposition of the eigensystem of the constant coefficient EULER-LAGRANGE equations.

For the optimal regulator, these equations take the form:

$$\begin{bmatrix} \dot{x} \\ \dot{\lambda} \end{bmatrix} = \begin{bmatrix} F & G^T B^{-1} G \\ A & -F^T \end{bmatrix} \begin{bmatrix} x \\ \lambda \end{bmatrix}$$

For the optimal filter or estimator, the equations are of the form:

$$\begin{bmatrix} \dot{x} \\ \dot{\lambda} \end{bmatrix} = \begin{bmatrix} F & G \otimes Q \otimes G^T \\ H^T R^{-1} H & -F^T \end{bmatrix} \begin{bmatrix} x \\ \lambda \end{bmatrix}$$

CFTSYSX, and all earlier program versions, use the method of eigenvector decomposition of the EULER-LAGRANGE equations described in [Ref. 7] for quadratic synthesis of control systems. Program calculations are based on the QR algorithm of Francis, modified by Wilkinson [Ref. 8]. Important features of this method of eigensystem analysis are its improved accuracy, and its insensitivity to widely separated eigenvalues.

A more detailed description of the QR algorithm and its numerical applications to eigensystem analysis may be found in [Ref. 8].

E. PROGRAM OVERVIEW

CFTSYSX and its 41 subroutines may be divided into three basic categories:

- 1) setup and computation sequencing
- 2) data input
- 3) calculation

A brief and general description of the program modules and subroutines supporting these basic categories concludes this section.

1. Problem Description/Program Flow Control

The MAIN program allows interactive selection of all program flow control flags, and is aided by three subroutines : RDREAL, RDINI, and RDCHAR. A detailed description of these subroutines is provided in Chapter III. Subroutine CHECK verifies the consistency of all requested program options.

2. Interactive Data Input

Interactive data input is performed by the 14 READ_ subroutines. A detailed description of the operation of these subroutines is also included in Chapter III. Internal data generation or external data input is provided by subroutine SETUP.

3. Calculation Sequencing

Subroutine INNER functions as a second MAIN program. It orders the data input/calculation sequences for the type of problem being solved and performs numerous matrix calculations. It is from this subroutine that all input and calculation sequences are ordered and performed.

4. QR Algorithm Transformation

Subroutines MINV, BALANC, ORTHES, ORTRAN, HQF, HQF2, BALBAK, CNCRM, and ERExit perform the major calculation sequences of the "QR" algorithm.

5. Riccati Equation Calculations

Subroutine FGAIN separates the eigenvalues and eigenvectors of the Euler-Lagrange equations by eigenvector decomposition. RGAIN and subroutine SCOV calculate the steady-state solution of the Riccati equations for the controller or estimator problem. Subroutine SCOV computes

the covariance matrix solution to the algebraic Riccati equation.

6. Modal Calculations

Subroutine MCDE computes the modal transformations required for modal analysis.

7. Transfer Function Calculations

Subroutines TF, POLES, ZEROS, RESID, ACOMP, CCOMP, EQR, and Function SCI perform transfer function computations associated with Modal calculation sequences. Subroutine ESDCAI computes the power spectral density of the outputs or controls of a controlled system.

8. Data Output

Subroutine RAERNT prints all program calculations in object time format. Subroutine MATPRT allows variable format screen viewing of all interactive matrix data input.

III. INTERACTIVE PROGRAM OPERATION

A. DESIGN CONSIDERATIONS

During the development of CPTSYSX, all program modifications and additions were focused primarily toward interactive user operation. Experience has demonstrated that interactive computer communication offers many advantages in the research and problem solving environment. The opportunity for flexible and immediate computer communication, as well as the ability to select alternate solution methods, are significant advantages to the user; advantages which are unavailable in a batch processing environment.

Although previous versions of OPTSYS produced all the desired calculations in the batch environment, the input format and data sequencing and naming conventions were confusing to many users. The user was burdened with the necessity of verifying the correctness of input data format and program flow control flag settings for each program run, in order to ensure the desired calculation sequence was properly performed.

These requirements, combined with incomplete program documentation, prompted a lack of confidence in the results and discouraged many potential users from ever attempting to use CPTSYS.

B. PROGRAM LANGUAGE

CPTSYSX is programmed in FORTRAN IV, following the conventions of IBM System/360/370 FORTRAN IV language. Very few program features have been incorporated which are not written in ANSI Standard FORTRAN. These subtle differences notwithstanding, OPTSYSX has been compiled and run

under both FORTRAN IV (G1) and FORTRAN H (Extended) compilers on the IBM 3033. Although the overall program length is in excess of 2800 lines of text, it is considered completely portable from one operating system to another.

On the presumption that most scientific and research personnel are familiar with FORTRAN language, program modifications may be easily undertaken once system operation is understood.

C. GENERAL PROGRAM MODIFICATIONS

All of the previously developed numerical calculation sequences of OPTSYS were retained un-modified in OPTSYSX.

Those program sequences requiring the input of diagonal cost or covariance matrix elements were deleted or modified to improve user flexibility in entering any desired weighting matrix, diagonal or otherwise. This modification streamlined program operation through elimination of several program flow control flags; reduced a measure of user uncertainty; and further decreased the required degree of user program familiarity--promoting uninterrupted operation.

The FEAD subroutines represent variations on the principle of simple, effective methods of interactive input, coupled with error-correction/recovery sequences.

Subroutines RDCHAR, RDINT, and RDREAL were developed by the author and Cdr. P.D. Sullivan to accomodate varying input format requirements; null-string entry protection was developed by Cdr. P.D. Sullivan. These program features are discussed in greater detail later in this chapter.

E. INTERACTIVE PROGRAM DEVELOPMENT

1. Program Flow Control

Initial program development required an understanding of the various problem descriptions as well as program input and calculation sequence. After careful analysis of [Ref. 2], a basic program flow control diagram was established. From this flow control diagram, a logical branching network was formulated whose path could be determined through either binary logic or numerical selection.

Three basic branching categories were established from the various problem description statements:

Logical-----{"Yes" or "No"..."}

Integer-----{"1", "2", etc..."}

Real Number----{"Input the value of..."}

From the viewpoint of free-format computer communication, integer and real number input presented no significant problems. FORTRAN compiler language is written such that numerical data input (real number or integer) is expected, thereby requiring only an INTEGER or REAL data type statement within the program. Once the data type has been declared, the desired values may then be input with a free-format READ (5,*) statement.

One note of caution concerning numerical data input in free-format deserves mentioning: Although all FORTRAN compilers treat character string input as an illegal data type conversion, many will automatically convert the inadvertent character entry to a "zero" and continue execution. Protection against inadvertent errors of this type is discussed later in this chapter.

Logical input ("Yes" or "No") poses a unique problem to programmers. The usual method of incorporation is to require the user to convert the logical answer into an integer i.e., "Yes" = 1, "No" = 2. These integers may then be read directly into the program, determining program flow.

Although this method may promote programming ease, it requires the user to adopt an unnatural habit pattern--one which increases the possibility of abnormal program termination in the event of inadvertent user error.

A more refined (from a programming language standpoint) and ergonomic (from the user viewpoint) method of logical selection involves utilization of the entire character string answer as an input value. This method has been incorporated into OPTSYSX.

The logical strings ("Yes" and "No") are declared as character strings in a data type statement within the program or subprogram. A format statement is also included in the program or subprogram utilizing the "A"-field to specify the desired character field width. A REWIND statement is then incorporated in the specific program or subprogram immediately prior to each logical string input point. This REWIND statement allows the input device (the terminal screen in this case) to read the character string in the same manner as free-format data input. The character field width for this modification was established at A1, allowing streamlined operation with the user typing either "Y" or "Yes" for an affirmative reply; "N" or "No" for a negative reply.

2. General Input Sequencing Requirements

All data input to OPTSYSX is in matrix or vector format. This data input must be correlated in accordance with the problem description and then properly sequenced in order for the program to perform the desired calculations.

The original and modified OPTSYS programs [Ref. 2] and [Ref. 3] required not only problem description knowledge but complete user familiarity with the detailed calculation sequence of the program. The latter point was considered a significant disadvantage. Elimination of this disadvantage

was an area where interactive programming offered the greatest benefits to the potential user; and it was toward this end that the remaining modifications of OPTSYSX were directed.

3. Interactive Data Input

In its calculation sequences, OPTSYSX requires the input of up to 14 unique matrices or vectors. Once the previously described program flow control diagram was constructed, data entry points for each matrix or vector were established. At each of the 16 program data entry points, the required input matrix or vector was determined. Fourteen input subroutines were added to the original program in order to accomodate interactive data entry.

These matrix input subroutines were written such that the user is first informed which specific matrix or vector is required; then prompted for the individual matrix element values. These values are then individually and sequentially entered from the terminal. Once the matrix or vector is filled, it is returned to the terminal screen for user verification and correction if necessary.

Interactive sequential data entry was programmed by means of a two-dimensional DO loop, with a terminal prompt of the matrix name and element position prior to the element value entry. Data element input is via a free-format READ (5,*) program statement.

Once the matrix data entry sequence is complete, that input matrix is returned to the terminal screen in variable format for user ease in row identification. With an arbitrary data field width of 12 characters, the maximum number of matrix elements available on an 80 column terminal screen is six. Provided the matrix column dimension is less than six, this restriction presented no programming format limitations.

For those matrices whose column dimension exceeded six, elements were progressively written on subsequent terminal lines. Once the matrix row is filled the screen is double-spaced, and element display is repeated in the same fashion. This method allows the user to view the matrix much as he would expect to see it, providing the advantage of ease in row and column identification.

Within CPTSYSX, subroutine MATPRT performs this variable-format print sequence. The print sequence was arbitrarily terminated with a matrix size of 16 X 16, presuming that users with larger systems of equations would select alternate forms of data input.

4. Saving Interactive Input

In most control system design problems, the system matrices generally remain relatively unchanged for a desired sequence of design calculations.

In order to relieve the user of the burden of repeated system entry in the interactive mode, several additional program flow control flags were added, allowing the option of saving the entire original system of matrices for subsequent calculations. Separate options for saving each system matrix are automatically offered at the end of each program run.

These matrix saving options provide a further advantage to the user in that the matrices are redisplayed for verification prior to calculation execution. Users may then change individual matrix elements, relieved of the burden of full system re-entry.

5. User-Defined Input Files

Although the basic objective of this work is to provide the user with a totally interactive method of data input, several disadvantages to the method of individual

matrix element input are apparent--input of very large matrices is unwieldy and time-consuming; input of systems of matrices whose elements remain unchanged from run to run is inefficient.

In order to provide an increased measure of user flexibility in data input, subroutine SETUP was modified to include provisions for matrix data input from a data file on the user's disk. The three system matrices, [F], [G], and [GAMMA] may now be input from the user's disk. Minor program modification is required of the user as follows:

- a. FRICMS Filedef commands must be modified or added to reflect the name and location of each data set.
- b. The READ Format statement (or statements) must be changed to reflect the proper data format of the user's input file.

6. Internal Data Generation

As a further measure of flexibility, the documentation within subroutine SETUP was expanded to include several specific examples of internal matrix data generation. The three system matrices [F], [G], and [GAMMA] may be generated either within user-written two-dimensional DO-loops or by direct assignment statements. These methods may be preferable for the input of very large matrices with few non-zero elements.

A specific example of internal program generation of the output equation [H] matrix is included in subroutine READH. This matrix input method may be preferable for the entry of a large output equation matrix with very few non-zero elements.

Once these modifications have been made to subroutine SETUP or subroutine READH (as desired), the program should be re-compiled and then run in the usual manner. An

interactive program prompt is provided at the beginning of OPTSYSX offering the user the option of specifying the desired method of data input.

OPTSYSX was further modified to include the ability to input the [H] matrix (or other required input matrices) from separate data files. Users with rudimentary programming skills may now modify subroutine READH (or one of the other specific READ subroutines) in the manner previously described for subroutine SETUP or subroutine READH. Detailed examples of the nature and extent of these modifications may be found in Appendix A.

7. Data Entry Correction

In an effort to protect users from errors in data input, an error correction sequence was incorporated into each matrix input subroutine.

Once the entire matrix or vector is displayed on the terminal screen the user is prompted with the question, "Do you wish to change the value of any matrix element? Type 'Yes' or 'No'." If the user types "No", program execution continues.

If the user types "Yes", he is then prompted with three additional statements specifying the row number of the element to be changed, the column number of the element to be changed, and the value to be inserted into that matrix element. After the corrected value is entered, the new matrix values are returned to the screen for re-verification.

This correction sequence continues indefinitely until the user signifies that no additional changes are necessary. Program execution then proceeds normally.

E. USER-ERROR PROTECTION FEATURES

Many interactive computer programs suffer the striking characteristic of abnormal program termination (without recovery!) should the user inadvertently make an erroneous keyboard entry. Examples of these inadvertent errors include--entry of a keyboard character or character string when the program expects a numerical value; entry of a numerical value when the program is expecting a character string; entry of a null string. In order to preclude abnormal program termination due to these types of inadvertent user errors, several program protection features were incorporated into OPTSYSX.

1. Data Type Conversion Errors

Three subroutines--RDREAL, RDINT, RDCHAR--were added to OPTSYSX in order to ensure that the proper input data type is provided to the program. Subroutine RDREAL is called at any point a real number or zero integer input may be encountered; subroutine RDINT is called at any point a non-zero integer input is required; subroutine RDCHAR is called at any point a character string ("Yes" or "No") input is required.

Within each of these subroutines a null string entry protection loop was incorporated (allowing one recovery); prompting the user for the correct data type input, and returning an error message in the event an incorrect data type is encountered.

Within subroutine RDINT, improper data type entry was further protected by the addition of a three-way IF comparison of entry integer magnitude. This modification precludes illegal (but automatic, with some compilers) data type conversion errors.

These program design features boast the additional advantage of allowing normal program termination at any point in the data input phase by merely pressing the "Enter" key twice.

2. Inconsistent Program Control Flag Errors

Earlier versions of OPTSYS [Ref. 2] and [Ref. 3], did provide user error messages in the event of inconsistent program flow control flags, but terminated the program. This feature was undesirable from the standpoint of smooth interactive program operation, and was improved in OPTSYSX.

Subroutine CHECK was modified to include RETURN statements any time inconsistent program flow control flags are encountered. The user is notified of the type of error encountered; that run termination has occurred; and prompted regarding his desire to return to the beginning of the program or terminate execution completely.

IV. PROGRAM USE AND EXAMPLES

This chapter contains several basic examples of the numerous types of problems which may be solved using CPTSYSX. Included with these examples are copies of each recorded terminal session.

Potential users should examine carefully the example of program failure found in Section D. This example clearly demonstrates that unstable modes or incorrect choice of certain design parameters may cause program failure (and incorrect output!), even with the highly stable "QR" algorithm. It also indicates one possible method of correcting this type of failure by merely making a very small change to one of the design parameters.

A. OPEN-LCCP EIGENSYSTEM ANALYSIS

The following open-loop eigenvalue example was taken from [Ref. 9, p.669].

Examination of the following program output shows open-loop eigenvalues at -1, -2, and -3. Note that the eigenvectors of the left and right eigenvector matrices (pa. 33) correspond in column fashion to the open-loop eigenvalues calculated immediately above them (pa. 32).

The full terminal session is recorded below, with user input in lower case letters following each "?" .

```
record on
BEGIN RECORDING OF TERMINAL SESSION
R; T=0.01,0.02 21:49:30
filedef 06 term (recfm fa blksize 133
global txtlib fortmod2 mcd2eeh imsldp ncnimsl
load cptsysx (start
```


EXECUTION BEGINS...

CPTSYSX IS A COMPLETELY INTERACTIVE OPTIMAL SYSTEMS CONTROL PROGRAM. IT WILL SOLVE NUMEROUS CONTROL PROBLEMS ON THE FOLLOWING TYPES OF SYSTEMS CONTROL EQUATIONS:

XDOT = (F)*X + (G)*U + (GAM)*(W+W0)

MEASUREMENT EQUATION--

Z = (H)*X + (D)*W

REGULATOR PERFORMANCE INDEX--

J = 1/2 * INTEGRAL (Yt*(A)*Y + Ut*(B)*U) DT

STATE FEEDBACK GAIN DEFINITION--

U = -(C)*X

DO YOU WISH TO CONTINUE? TYPE "YES" OR "NO".

yes

--DATA ENTRY--

ALTHOUGH OPISYSX IS SPECIFICALLY DESIGNED TO READ ALL MATRIX DATA INTERACTIVELY, SEVERAL ALTERNATE METHODS ARE AVAILABLE TO USERS:

METHOD 1--THE "F", "G", AND "GAMMA" MATRICES
MAY BE READ FROM SEPARATE DATA FILES.

METHOD 2--THE "F", "G", AND "GAMMA" MATRICES MAY BE
EXPLICITLY DEFINED WITHIN SUBROUTINE "SETUP".

(NOTE: IN EITHER CASE, THE USER SHOULD OBTAIN A COPY
OF THE PROGRAM LISTING AND EXAMINE
THE EXAMPLES CONTAINED IN S/R "SETUP".)

DO YOU WISH TO CONTINUE? TYPE "YES" OR "NO".

yes

DO YOU WISH TO INPUT THE "F", "G", AND "GAMMA"
MATRICES FROM SUBROUTINE "SETUP" IAW THE
METHOD DESCRIBED ON THE PREVIOUS SCREEN?
TYPE "YES" OR "NO".

no

GENERAL OPISYSX OPTIONS:

OPTION 1 -- SYSTEM ANALYSIS WITHOUT
OPEN-LOOP EIGENSYSTEM CALCULATIONS.

OPTION 2 -- SYSTEM ANALYSIS WITH OPEN-LOOP

EIGENSYSTEM CALCULATIONS.

CPTION 3 -- CEEN-LOCF EIGENSYSTEM FOUND
AND PROGRAM TERMINATES.

("F"-MATRIX ENTRY FOLLOWS IMMEDIATELY.)

CPTION 4 -- MCAL DISTRIBUTION MATRICES COMPUTED
WITHOUT FILTER OR REGULATOR SYNTHESIS
OR STEADY-STATE ANALYSIS.

SELECT AN OPTICN: 1,2,3, OR 4.

?

3

ENTER THE # OF STATES (NS) OF THE SYSTEM MATRIX
"F"-MATRIX .

?

3

FLAG/PARAMETER SETTINGS FOR THIS RUN ARE AS FOLICWS:

IOL	IC	IR	ISS	IM	ITF1	ITF2	ITF3	IFDFW	IE	IDEBUG
2	0	0	0	C	0	0	0	0	0	0
ISET	IDSTAE	IPSD	IYU	INCRM	IREG	NS	NC	NOB	NG	
0	0	0	C	0	0	3	0	0	0	

ORDER OF SYSTEM = 3

NUMBER OF CONTROLS = 0

NUMBER OF OBSERVATIONS = 0

NUMBER OF PROCESS NOISE SOURCES = 0

ENTER THE SYSTEM MATRIX "F"-MATRIX

DIMENSION = # STATES (NS) X # STATES (NS)

THE ELEMENT F(1, 1)=

?

0

THE ELEMENT F(1, 2)=

?

1

THE ELEMENT F(1, 3)=

?

0

THE ELEMENT F(2, 1)=

?

C

THE ELEMENT F(2, 2)=

?

0

THE ELEMENT F(2, 3)=

?

1

THE ELEMENT F(3, 1)=

?

-6

THE ELEMENT F(3, 2)=

?

-11

THE ELEMENT F(3, 3)=

?

-6

THE SYSTEM MATRIX "F"-MATRIX ...

0.0	1.00000	0.0
0.0	0.0	1.00000
-6.00000	-11.00000	-6.00000

DO YOU WISH TO CHANGE THE VALUE OF ANY MATRIX ELEMENT?

TYPE "YES" OR "NO".

N

OPEN LCCP DYNAMICS MATRIX.....F..

0.0	0.1000E+01	0.0
0.0	0.0	0.1000D+01
-0.6000E+01	-0.1100E+02	-0.6000D+01

OPEN IOCP EIGENVALUES.....DET(SI-F)..

-1.00000E+00:-2.00000E+00:-3.00000D+00:

OPEN ICCF RIGHT EIGENVECTOR MATRIX.....T....

-5.773503D-01 -2.182179D-01 1.048285D-01

5.773503D-01 4.364358D-01 -3.144355D-01

-5.773503D-01 -8.728716D-01 9.434564D-01

OPEN ICCF LEFT EIGENVECTOR MATRIX.....T-INV..

-5.196152D+00 -4.330127D+00 -8.660254D-01

1.374773D+01 1.833030D+01 4.582576D+00

9.539392D+00 1.430909D+01 4.769696D+00

ANALYSIS COMPLETE. DO YOU WANT ANOTHER RUN?

TYPE "YES" OR "NO".

DO

.....CPTSYSX IS NOW TERMINATED.....

R; T=0.28,0.89 21:52:08

record off

END RECORDING OF TERMINAL SESSION

E. REGULATOR SYNTHESIS

The following regulator synthesis example was taken from "Lecture Notes on Advanced Control Systems", by Professor D.J. Collins of the Naval Postgraduate School, Monterey, Ca. This example involved determination of the optimal regulator gains based on an arbitrarily chosen quadratic index; with the various system and cost matrices described below.

Examination of the extensive program output indicates that the optimal regulator gains are: -5.0 and -SQRT(10.0). The algebraic sign of the gains is consistent with the definition displayed on the first screen of the program (p. 34).

The full terminal session is recorded below, with user input in lower case letters following each "?" .

record cr

BEGIN RECORDING OF TERMINAL SESSION

R; I=C.01,C.02 13:55:26

filedef C6 term (recfm fa blksize 133
global titlib fortmod2 mod2eeh imslp nonimsl
load optsysx (start
EXECUTION BEGINS...

OPTSYSX IS A COMPLETELY INTERACTIVE OPTIMAL SYSTEMS CONTROL PROGRAM. IT WILL SOLVE NUMEROUS CONTROL PROBLEMS ON THE FOLLOWING TYPES OF SYSTEMS CONTROL EQUATIONS:

XDOT = (F)*X + (G)*U + (GAM)*(W+W0)

MEASUREMENT EQUATION--

Z = (H)*X + (D)*W + V

REGULATOR PERFORMANCE INDEX--

J = 1/2 * INTEGRAL (Yt*(A)*Y + Ut*(B)*U) DT

STATE FEEDBACK GAIN DEFINITION--

U = -(C)*X

DO YOU WISH TO CONTINUE? TYPE "YES" OR "NO".

Y

--DATA ENTRY--

ALTHOUGH OPTSYSX IS SPECIFICALLY DESIGNED TO READ ALL MATRIX DATA INTERACTIVELY, SEVERAL ALTERNATE METHODS ARE AVAILABLE TO USERS:

METHOD 1--THE "F", "G", AND "GAMMA" MATRICES

MAY BE READ FROM SEPARATE DATA FILES.

METHOD 2--THE "F", "G", AND "GAMMA" MATRICES MAY BE EXPLICITLY DEFINED WITHIN SUBROUTINE "SETUP".

(NOTE: IN EITHER CASE, THE USER SHOULD OBTAIN A COPY OF THE PROGRAM LISTING AND EXAMINE THE EXAMPLES CONTAINED IN S/R "SETUP".)

DO YOU WISH TO CONTINUE? TYPE "YES" OR "NO".

Y

DO YOU WISH TO INPUT THE "F", "G", AND "GAMMA" MATRICES FROM SUBROUTINE "SETUP" IAW THE

METHOD DESCRIBED ON THE PREVIOUS SCREEN?

TYPE "YES" OR "NO".

N

GENERAL CPTSYSX OPTIONS:

CETION 1 -- SYSTEM ANALYSIS WITHOUT
OPEN-LOOP EIGENSYSTEM CALCULATIONS.
CETION 2 -- SYSTEM ANALYSIS WITH OPEN-LOOP
EIGENSYSTEM CALCULATIONS.
CETION 3 -- OPEN-LOOP EIGENSYSTEM FOUND
AND PECGFAM TERMINATES.
("F"--MATRIX ENTRY FOLLOWS IMMEDIATELY.)
CETION 4 -- MODAL DISTRIBUTION MATRICES COMPUTED
WITHOUT FILTER OR REGULATOR SYNTHESIS
OR STEADY-STATE ANALYSIS.
SELECT AN OPTION: 1, 2, 3, OR 4.

?

2

DO YOU DESIRE RMS VALUES OF STATE AND CONTROL?
TYPE "YES" OR "NO".

NO

CPTSYSX LQR/CLASSICAL OPTIONS:

CETION 1 -- OPTIMAL FILTER AND/OR REGULATOR
SYNTHESIS WITH NO EXTERNAL "C" OR "K"
MATRIX INPUT.
CETION 2 -- OPTIMAL FILTER AND/OR REGULATOR
SYNTHESIS WITH EXTERNAL "C"
MATRIX INPUT.
CETION 3 -- OPTIMAL FILTER AND/OR REGULATOR
SYNTHESIS WITH EXTERNAL "K"
MATRIX INPUT.
CETION 4 -- OPTIMAL FILTER AND/OR REGULATOR
SYNTHESIS WITH EXTERNAL "C" AND "K"
MATRIX INPUT.

SELECT AN CETION: 1, 2, 3, OR 4.

?

1

DO YOU WISH TO DETERMINE THE STEADY-STATE RESPONSE
FOR A CONSTANT DISTUREANCE?

TYPE "YES" OR "NO".

DO

DO YOU WISH TO DETERMINE THE MODAL DISTRIBUTION
AND GAIN MATRICES?

TYPE "YES" OR "NO".

DO

OPEN-LOOP TRANSFER FUNCTION OPTIONS:

OPTION 1 -- NC OPEN-LOOP TRANSFER FUNCTIONS COMPUTED.

OPTION 2 -- POLES, RESIDUES, AND ZEROS COMPUTED.

OPTION 3 -- ONLY POLES AND ZEROS COMPUTED.

OPTION 4 -- ONLY POLES AND RESIDUES COMPUTED.

SELECT AN OPTION: 1, 2, 3, OR 4.

?

1

NOISE TRANSFER FUNCTION OPTIONS:

OPTION 1 -- NC NOISE TRANSFER FUNCTIONS COMPUTED.

OPTION 2 -- POLES, RESIDUES, AND ZEROS COMPUTED.

OPTION 3 -- ONLY POLES AND ZEROS COMPUTED.

OPTION 4 -- ONLY POLES AND RESIDUES COMPUTED.

SELECT AN OPTION: 1, 2, 3, OR 4.

?

1

COMPENSATOR TRANSFER FUNCTION OPTIONS:

OPTION 1 -- NC COMP. TRANSFER FUNCTIONS COMPUTED.

OPTION 2 -- POLES, RESIDUES, AND ZEROS COMPUTED.

OPTION 3 -- ONLY POLES AND ZEROS COMPUTED.

OPTION 4 -- ONLY POLES AND RESIDUES COMPUTED.

(NOTE: A COMPENSATOR TRANSFER FUNCTION CAN BE
COMPUTED ONLY IF BOTH A REGULATOR AND
FILTER ARE SYNTHESIZED AND/CR INPUT.)

SELECT AN OPTION: 1, 2, 3, OR 4.

?

1

WILL A FEED-FORWARD DISTRIBUTION MATRIX

("D" - MATRIX) BE INPUT ?

TYPE "YES" OR "NO".

NO

THIS OPTION DETERMINES THE CRITERIA FOR DECIDING WHEN A MARKOV PARAMETER IS ZERO-THE MARKOV PARAMETER INDICATES THE ORDER OF THE NUMERATOR POLYNOMIAL OF EACH TRANSFER FUNCTION.

ALL "N" ZEROS OF THIS POLYNOMIAL ARE PRINTED OUT AND THIS TEST TELLS HOW MANY EXTRA ROOTS EXIST AT Z = 0. LESS THAN 10.0** (-IE) IS CONSIDERED ZERO.

THE DEFAULT VALUE OF THIS PARAMETER (IE) IS 6.

IN OTHER WORDS, IE = 1.0E-6.

IF YOU DESIRE A DIFFERENT MARKOV CRITERIA, TYPE THE INTEGER VALUE.

IF YOU DESIRE THE DEFAULT VALUE, TYPE "0" (ZERO)

?

C

DO YOU DESIRE TO SYNTHESIZE A STABLE FILTER (OR REGULATOR) BY DESTABILIZING THE ORIGINAL SYSTEM?

(NOTE: WORKS FOR FILTER OR REGULATOR BUT NOT FOR BOTH IN THE SAME RUN.)

TYPE "YES" OR "NO".

NO

DO YOU DESIRE TO PRINT THE EULER-LAGRANGE EIGENSYSTEM FOR THE DECOMPOSITION (FOR CHECKING THE PROGRAM)?

TYPE "YES" OR "NO".

yes

POWER SPECTRAL DENSITY (PSD) OPTION 1 :

OPTION 1 -- COMPUTE THE PSD OF THE OUTPUTS AND/OR THE CONTROLS OF THE CONTROLLED SYSTEM WHEN FORCED BY PROCESS AND MEASUREMENT NOISE.

(NOTE: BOTH A REGULATOR AND A FILTER MUST BE

RESIDENT IN THE PROGRAM TO USE THIS OPTION.)

OPTION 2 -- SAME AS OPTION 1 ABOVE BUT ONLY PRINT THE RESIDUES OF EACH TRANSFER FUNCTION USED IN THE PSD COMPUTATION.

OPTION 3 -- NOT DESIRED.

SELECT AN OPTION: 1, 2, OR 3.

?

3

DO YOU DESIRE REGULATOR SYNTHESIS ONLY?

TYPE "YES" OR "NO".

yes

ENTER THE # OF STATES (NS) OF THE SYSTEM MATRIX
("F"-MATRIX).

?

2

ENTER THE # OF CONTROLS (NC) OF THE CONTROL SYSTEM MODEL
("G"-MATRIX).

?

1

ENTER THE # OF MEASUREMENTS OR OBSERVATIONS (NO) OF THE
("H"-MATRIX).

?

2

ENTER THE # OF PROCESS NOISE SOURCES (NG) OF THE
("GAMMA"-MATRIX).

?

C

FLAG/PARAMETER SETTINGS FOR THIS RUN ARE AS FOLLOWS:

IOL	IQ	IR	ISS	IM	ITF1	ITF2	ITF3	IFDFW	IE	IDEBUG
1	0	C	0	C	0	0	0	0	0	1

ISET	IDSTAE	IPSD	IVU	INCIM	IREG	NS	NC	NOB	NG
0	0	0	C	0	1	2	1	2	0

ORDER OF SYSTEM = 2

NUMBER OF CONTROLS = 1

NUMBER OF OBSERVATIONS = 2

NUMBER OF PROCESS NOISE SOURCES = 0

ENTER THE SYSTEM MATRIX "F"-MATRIX

DIMENSION = # STATES (NS) X # STATES (NS)

THE ELEMENT F(1, 1) =


```

?
0
    THE ELEMENT F( 1, 2)=
?
1
    THE ELEMENT F( 2, 1)=
?
0
    THE ELEMENT F( 2, 2)=
?
C
        THE SYSTEM MATRIX "F"-MATRIX ...
C.0      1.00000
C.0      0.0
DO YOU WISH TO CHANGE THE VALUE OF ANY MATRIX ELEMENT?
    TYPE "YES" OR "NO".
NO
    CFEN LCCP DYNAMICS MATRIX.....F..
0.0      0.1000D+01
0.0      0.0
    CFEN LCCP EIGENVALUES.....DET(SI-F)..
0.0      : 0.0      :
    CFEN LCCP RIGHT EIGENVECTOF MATRIX.....T....
1.000000D+00 -1.000000D+00
0.0      2.220446D-16
    CFEN LCCP LEFT EIGENVECTOR MATRIX.....T-INV..
1.000000D+00 4.503600D+15
0.0      4.503600D+15
ENTER THE MEASUREMENT SCALING MATRIX "H"-MATRIX .
    DIMENSION = # OBSERVATIONS (NO) X # STATES (NS)
    THE ELEMENT H( 1, 1)=
?
1

```


THE ELEMENT H(1, 2)=

?
C

THE ELEMENT H(2, 1)=

?
C

THE ELEMENT H(2, 2)=

?
1

THE MEASUREMENT SCALING MATRIX "H"-MATRIX ...

1.00000 0.0

0.0 1.00000

DO YOU WISH TO CHANGE THE VALUE OF ANY MATRIX ELEMENT?

TYPE "YES" OR "NO".

NO

MEASUREMENT SCALING MATRIX.....H..

0.1000E+01 0.0

0.0 0.1000E+01

ENTER THE OUTPUT MEASUREMENT COST MATRIX "A"-MATRIX .

DIMENSION = # OBSERVATIONS (NO) X # OBSERVATIONS (NO)

THE ELEMENT A(1, 1)=

?
25

THE ELEMENT A(1, 2)=

?
C

THE ELEMENT A(2, 1)=

?
0

THE ELEMENT A(2, 2)=

?
C

THE OUTPUT MEASUREMENT COST MATRIX "A"-MATRIX ...

25.00000 0.0

0.0 0.0

DO YOU WISH TO CHANGE THE VALUE OF ANY MATRIX ELEMENT?
TYPE "YES" CR "NO".

10

CUTPUT COST MATRIX.....A..

0.2500E+02 0.0

0.0 0.0

ENTER THE CONTROL DISTRIBUTION MATRIX "G"-MATRIX .

DIMENSION = # STATES (NS) X # CONTRCLS (NC)

THE ELEMENT G(1, 1)=

?

0

THE ELEMENT G(2, 1)=

?

1

THE CONTROL DISTRIBUTION MATRIX "G"-MATRIX ...

0.0

1.00000

DO YOU WISH TO CHANGE THE VALUE OF ANY MATRIX ELEMENT?
TYPE "YES" CR "NO".

10

ENTER THE CONTROL COST WEIGHTING MATRIX "B"-MATRIX

DIMENSION = # CONTRCLS (NC) X # CONTROLS (NC)

THE ELEMENT B(1, 1)=

?

1

THE CONTROL COST MATRIX.....B...

1.00000

DO YOU WISH TO CHANGE THE VALUE OF ANY MATRIX ELEMENT?
TYPE "YES" CR "NO".

10

THE CONTROL DISTRIBUTION MATRIX.....G..

0.0

0.1000E+01

THE CONTROL COST MATRIX.....B..

0.10000E+01

EULER-LAGRANGE SYSTEM MATRIX...

0.0	1.000000D+00	0.0	0.0
0.0	0.0	0.0	-1.000000D+00
-2.500000D+01	0.0	0.0	0.0
0.0	0.0	-1.000000D+00	0.0

EIGENVALUES AND EIGENVECTORS OF THE 2N X 2N

EULER-LAGRANGE SYSTEM AFTER HQR2.....

-1.581139D+00	1.581139D+00		
-1.581139D+00	-1.581139D+00		
1.581139D+00	1.581139D+00		
1.581139D+00	-1.581139D+00		
-7.430443D-02	7.168812D-02	-1.824925D-01	-1.503482D-01
4.136755D-03	-2.308345D-01	-5.082459D-02	-5.262675D-01
-1.154172D+00	-2.068377D-02	2.631337D+00	-2.541229D-01
-3.584406D-01	-3.715222D-01	-7.517412D-01	9.124627D-01

EIGENSYSTEM OF OPTIMAL REGULATOR.....

EIGENVECTORS FROM RGAINE PRIOR TO CNORM

-1.459925D-01	-2.616313D-03
2.349712D-01	-2.266977D-01

C-LCCP OPTIMAL REG. E-VALUES...DET(SI-F+G*C) ..

-1.58114E+00, 1.58114E+00:

C-LCCP EIGHT EIGENVECTOR MATRIX.....M....

-3.162278D-01	-3.162278D-01
1.000000D+00	0.0

CONTROL EIGENVECTOR MATRIX.....C*M..

-1.581139D+00 1.581139D+00

C-ICCF CPT. REG. LEFT E-VECTOR MATRIX..M-INV..

0.0 1.000000D+00
-3.162278D+00 -1.000000D+00

THE OPTIMAL FEEDBACK GAIN CONTROL MATRIX...C=5.162278D+00...
-5.00001D+00 -3.1623D+00

THE CLOSED LOOP DYNAMICS MATRIXF=G*C..

0.0 1.000000D+00
-5.000000D+00 -3.162278D+00

ANALYSIS COMPLETE. DO YOU WANT ANOTHER RUN?

TYPE "YES" OR "NO".

no

.....OPTSYSX IS NOW TERMINATED.....

F; T=C.42/1.85 14:03:03

recrd off

END FECCFDING OF TERMINAL SESSION

C. FILTER SYNTHESIS

The following Kalman filter synthesis example was taken from "Lecture Notes on Advanced Control Systems", by Professor D.J. Collins of the Naval Postgraduate School, Monterey, Ca.

This example involved determination of the optimal filter gains of an arbitrary system; modeled nearly identically to the previous regulator problem.

In its present configuration, OPTSYSX program sequencing requires the design of an optimal regulator, prior to performing any optimal estimator synthesis. In order to comply with built-in program sequencing conventions, and circumvent program difficulties which may not be specified in the particular system model, optimal filter synthesis may be accomplished by entering the identity matrix [I] in these

program input sequences requiring the entry of an output cost (weighting) matrix. Although the optimal regulator calculations may differ from those expected, the optimal estimator calculations will be correct for the system model.

Examination of the extensive program output indicates that the optimal filter gains are: -5.0, and -SQRT(2.0).

The full terminal session is recorded below, with user input in lower case letters following each "?" .

```
record on
BEGIN RECORDING OF TERMINAL SESSION
S; T=0.01,0.02 21:49:30
filedef 06 term (recfm fa blksize 133
global txtlib fortmod2 mcd2eeh imslidp nchimsl
lcad cptsyss (start
EXECUTION BEGINS...
OPTSYSX IS A COMPLETELY INTERACTIVE OPTIMAL SYSTEMS CONTROL
PROGRAM. IT WILL SOLVE NUMEROUS CONTROL PROBLEMS ON THE
FOLLOWING TYPES OF SYSTEMS CONTROL EQUATIONS:
XDOT = (F)*X + (G)*U + (GAM)*(W+W0)
MEASUREMENT EQUATION--
Z = (H)*X + (D)*W + V
REGULATOR PERFORMANCE INDEX--
J = 1/2 * INTEGRAL (Yt*(A)*Y + Ut*(B)*U) dt
STATE FEEDBACK GAIN DEFINITION--
U = -(C)*X
DO YOU WISH TO CONTINUE? TYPE "YES" OR "NO".
```

yes

```
-- DATA ENTRY--
ALTHOUGH OPTSYSX IS SPECIFICALLY DESIGNED TO READ
ALL MATRIX DATA INTERACTIVELY, SEVERAL ALTERNATE
METHODS ARE AVAILABLE TO USERS:
METHOD 1--THE "F", "G", AND "GAMMA" MATRICES
MAY BE READ FROM SEPARATE DATA FILES.
METHOD 2--THE "F", "G", AND "GAMMA" MATRICES MAY BE
```


EXPLICITLY DEFINED WITHIN SUBROUTINE "SETUP".
(NOTE: IN EITHER CASE, THE USER SHOULD OBTAIN A COPY
OF THE PROGRAM LISTING AND EXAMINE
THE EXAMPLES CONTAINED IN S/R "SETUP".)
DO YOU WISH TO CONTINUE? TYPE "YES" OR "NO".

yes

DO YOU WISH TO INPUT THE "F", "G", AND "GAMMA"
MATRICES FROM SUBROUTINE "SETUP" IAW THE
METHOD DESCRIBED ON THE PREVIOUS SCREEN?
TYPE "YES" OR "NO".

no

GENERAL CPTSYSX OPTIONS:

OPTION 1 -- SYSTEM ANALYSIS WITHOUT
OPEN-LOOP EIGENSYSTEM CALCULATIONS.
OPTION 2 -- SYSTEM ANALYSIS WITH OPEN-LOOP
EIGENSYSTEM CALCULATIONS.
OPTION 3 -- OPEN-LOOP EIGENSYSTEM FOUND
AND PROGRAM TERMINATES.
("F"-MATRIX ENTRY FOLLOWS IMMEDIATELY.)
OPTION 4 -- MCDAI DISTRIBUTION MATRICES COMPUTED
WITHOUT FILTER OR REGULATOR SYNTHESIS
OR STEADY-STATE ANALYSIS.
SELECT AN OPTION: 1,2,3, OR 4.

?

1

DO YOU DESIRE RMS VALUES OF STATE AND CONTROL?
TYPE "YES" OR "NO".

no

CPTSYSX LQE/CLASSICAL OPTIONS:

OPTION 1 -- OPTIMAL FILTER AND/OR REGULATOR
SYNTHESIS WITH NO EXTERNAL "C" OR "K"
MATRIX INPUT.
OPTION 2 -- OPTIMAL FILTER AND/OR REGULATOR
SYNTHESIS WITH EXTERNAL "C"
MATRIX INPUT.

OPTION 3 -- OPTIMAL FILTER AND/OR REGULATOR
SYNTHESIS WITH EXTERNAL "K"
MATRIX INPUT.

OPTION 4 -- OPTIMAL FILTER AND/OR REGULATOR
SYNTHESIS WITH EXTERNAL "C" AND "K"
MATRIX INPUT.

SELECT AN OPTION: 1, 2, 3, OR 4.

?
1

DO YOU WISH TO DETERMINE THE STEADY-STATE RESPONSE
FOR A CONSTANT DISTURBANCE?

TYPE "YES" OR "NO".

NO

DO YOU WISH TO DETERMINE THE MODAL DISTRIBUTION
AND GAIN MATRICES?

TYPE "YES" OR "NO".

NO

OPEN-LOOP TRANSFER FUNCTION OPTIONS:

OPTION 1 -- NO OPEN-LOOP TRANSFER FUNCTIONS COMPUTED.
OPTION 2 -- POLES, RESIDUES, AND ZEROS COMPUTED.
OPTION 3 -- ONLY POLES AND ZEROS COMPUTED.
OPTION 4 -- ONLY POLES AND RESIDUES COMPUTED.

SELECT AN OPTION: 1, 2, 3, OR 4.

?
1

NOISE TRANSFER FUNCTION OPTIONS:

OPTION 1 -- NO NOISE TRANSFER FUNCTIONS COMPUTED.
OPTION 2 -- POLES, RESIDUES, AND ZEROS COMPUTED.
OPTION 3 -- ONLY POLES AND ZEROS COMPUTED.
OPTION 4 -- ONLY POLES AND RESIDUES COMPUTED.
SELECT AN OPTION: 1, 2, 3, OR 4.

?
1

COMPENSATOR TRANSFER FUNCTION OPTIONS:

OPTION 1 -- NO COMP. TRANSFER FUNCTIONS COMPUTED.

CPTION 2 -- POLES, RESIDUES, AND ZEROS COMPUTED.

CPTION 3 -- ONLY POLES AND ZEROS COMPUTED.

CPTION 4 -- ONLY POLES AND RESIDUES COMPUTED.

(NOTE: A COMPENSATOR TRANSFER FUNCTION CAN BE
COMPUTED ONLY IF BOTH A REGULATOR
AND FILTER ARE SYNTHESIZED
AND/CR INPUT.)

SELECT ANY CPTION: 1, 2, 3, OR 4.

?

1

WILL A FEED-FORWARD DISTRIBUTION MATRIX
"D" - MATRIX BE INPUT ?

TYPE "YES" OR "NO".

NO

THIS CPTION DETERMINES THE CRITERIA FOR DECIDING WHEN
A MARKOV PARAMETER IS ZERO-THE MARKOV PARAMETER
INDICATES THE COEF OF THE NUMERATOR POLYNOMIAL OF EACH
TRANSFER FUNCTION.

ALL "N" ZEROS OF THIS POLYNOMIAL ARE PRINTED OUT AND
THIS TEST TELLS HOW MANY EXTRA ROOTS EXIST AT Z = 0.
LESS THAN 10.0**(-IE) IS CONSIDERED ZERO.

THE DEFAULT VALUE OF THIS PARAMETER (IE) IS 6.
IN OTHER WORDS, IE = 1.0E-6.

IF YOU DESIRE A DIFFERENT MARKOV CRITERIA,
TYPE THE INTEGER VALUE.

IF YOU DESIRE THE DEFAULT VALUE, TYPE "0" (ZERO)

?

C

DO YOU DESIRE TO SYNTHESIZE A STABLE FILTER (OR REGULATOR)
BY DESTABILIZING THE ORIGINAL SYSTEM?

(NOTE: WORKS FOR FILTER OR REGULATOR BUT NOT FOR BOTH
IN THE SAME RUN.)

TYPE "YES" OR "NO".

NO

DO YOU DESIRE TO PRINT THE EULER-LAGRANGE EIGENSYSTEM
STRUCTURE DECOMPOSITION (FOR CHECKING THE PROGRAM)?
TYPE "YES" OR "NO".

NO

POWER SPECTRAL DENSITY (PSD) OPTION 1 :

OPTION 1 -- COMPUTE THE PSD OF THE OUTPUTS AND/OR
THE CONTROLS OF THE CONTROLLED SYSTEM
WHEN FORCED BY PROCESS AND MEASUREMENT
NOISE. (NOTE: BOTH A REGULATOR AND A
FILTER MUST BE RESIDENT IN THE PROGRAM
TO USE THIS OPTION.)

OPTION 2 -- SAME AS OPTION 1 ABOVE BUT ONLY PRINT THE
RESIDUES OF EACH TRANSFER FUNCTION
USED IN THE PSD COMPUTATION.

OPTION 3 -- NOT DESIRED.

SELECT AN OPTION: 1, 2, OR 3.

?

3

DO YOU DESIRE REGULATOR SYNTHESIS ONLY?

TYPE "YES" OR "NO".

NO

ENTER THE # OF STATES (NS) OF THE SYSTEM MATRIX
"F"-MATRIX .

?

2

ENTER THE # OF CONTROLS (NC) OF THE SYSTEM MODEL
"G"-MATRIX .

?

1

ENTER THE # OF MEASUREMENTS OR OBSERVATIONS (NO)
"H"-MATRIX .

?

1

ENTER THE # OF PROCESS NOISE SOURCES (NG)
"GAMMA"-MATRIX .

?
0

FLAG/PARAMETER SETTINGS FOR THIS RUN ARE AS FOLLOWS:

IOL	IQ	IR	ISS	IM	ITF1	ITF2	ITF3	IFDFW	IE	IDBUG
0	0	C	0	0	0	0	0	0	0	0
ISET	IESTAE	IPSD	IYU	INORM	IREG	NS	NC	NOB	NG	
0	C	0	0	0	0	2	1	1	0	

ORDER OF SYSTEM = 2

NUMBER OF CONTROLS = 1

NUMBER OF OBSERVATIONS = 1

NUMBER OF PROCESS NOISE SOURCES = 0

ENTER THE SYSTEM MATRIX "F"-MATRIX

DIMENSION = # STATES (NS) X # STATES (NS)

THE ELEMENT F(1, 1) =

?

0

THE ELEMENT F(1, 2) =

?

1

THE ELEMENT F(2, 1) =

?

0

THE ELEMENT F(2, 2) =

?

C

THE SYSTEM MATRIX "F"-MATRIX ...

0.0 1.00000

0.0 0.0

DO YOU WISH TO CHANGE THE VALUE OF ANY MATRIX ELEMENT?

TYPE "YES" OR "NO".

NO

OPEN LCCP DYNAMICS MATRIX.....F..

0.0 0.1000E+01

0.0 0.0

ENTER THE MEASUREMENT SCALING MATRIX "H"-MATRIX .

DIMENSION = # OBSERVATIONS (NO) X # STATES (NS)

THE ELEMENT H(1, 1)=

?

1

THE ELEMENT H(1, 2)=

?

C

THE MEASUREMENT SCALING MATRIX "H"-MATRIX ...

1.00000 0.0

DO YOU WISH TO CHANGE THE VALUE OF ANY MATRIX ELEMENT?

TYPE "YES" OR "NO".

NO

MEASUREMENT SCALING MATRIX.....H..

0.1000E+01 0.0

ENTER THE OUTPUT MEASUREMENT COST MATRIX "A"-MATRIX .

DIMENSION = # OBSERVATIONS (NO) X # OBSERVATIONS (NO)

THE ELEMENT A(1, 1)=

?

1

THE OUTPUT MEASUREMENT COST MATRIX "A"-MATRIX ...

1.00000

DO YOU WISH TO CHANGE THE VALUE OF ANY MATRIX ELEMENT?

TYPE "YES" OR "NO".

NO

OUTPUT COST MATRIX.....A..

0.1000E+01

ENTER THE CONTROL DISTRIBUTION MATRIX "G"-MATRIX .

DIMENSION = # STATES (NS) X # CONTROLS (NC)

THE ELEMENT G(1, 1)=

?

0

THE ELEMENT G(2, 1)=

?
0

THE CONTROL DISTRIBUTION MATRIX "G"-MATRIX ...

0.0

0.0

DO YOU WISH TO CHANGE THE VALUE OF ANY MATRIX ELEMENT?
TYPE "YES" OR "NO".

yes

ENTER THE ROW NUMBER OF THE ELEMENT TO BE CHANGED.

?
2
1

ENTER THE COLUMN NUMBER OF THE ELEMENT TO BE CHANGED.

?
1

THE ELEMENT G(2, 1)=

?
1

THE CONTROL DISTRIBUTION MATRIX "G"-MATRIX ...

0.0

1.00000

DO YOU WISH TO CHANGE THE VALUE OF ANY MATRIX ELEMENT?
TYPE "YES" OR "NO".

no

ENTER THE CONTROL COST WEIGHTING MATRIX "B"-MATRIX

DIMENSION = # CONTROLS NC X # CONTROLS NC

THE ELEMENT B(1, 1)=

1
?

THE CONTROL COST MATRIX.....B...

1.00000

DO YOU WISH TO CHANGE THE VALUE OF ANY MATRIX ELEMENT?
TYPE "YES" OR "NO".

no

THE CONTROL DISTRIBUTION MATRIX.....G..
0.0
0.10001+01

THE CONTROL COST MATRIX.....B..
0.10001+01

EIGENSYSTEM OF OPTIMAL REGULATOR.....

C-LCCP OPTIMAL REG. E-VALUES...DET(SI-F+G*C) ..
-7.07107D-01, 7.07107D-01:

C-LCCP RIGHT EIGENVECTOR MATRIX.....M....
-7.071068D-01 -7.071068D-01
1.000000D+00 0.0

CONTROL EIGENVECTOR MATRIX.....C*M..
-7.071068D-01 7.071068D-01

C-LCCP OPT. REG. LEFT E-VECTOR MATRIX..M-INV..
0.0 1.000000D+00
-1.414214D+00 -1.000000D+00

THE OPTIMAL FEEDBACK GAIN CONTROL MATRIX...C=BINV*GT*S...
-1.00001+00 -1.414214D+00

THE CLOSED LOOP DYNAMICS MATRIXF-G*C..
0.0 1.000000D+00
-1.000000D+00 -1.414214D+00

ANALYSIS COMPLETE. DO YOU WANT ANOTHER RUN?
TYPE "YES" OR "NO".

NO

.....OPTSYSX IS NOW TERMINATED.....

F; T=0.54/2.84 15:36:49
reccrd off
END RECORDING OF TERMINAL SESSION

E. EXAMPLE OF PROGRAM FAILURE

The following pathological example of program failure during regulator synthesis was taken from the Journal of Guidance and Control, Vol.3, No.2, pp.190-192, March-April 1980.

In this example, the choice of the quadratic index value was the factor prompting program instability; leading to eventual program failure in subroutine HQR2. The calculated regulator gains of -5.1, and -3.1 (pa. 63) are not correct!

With a 'slight' modification of the cost matrix from a previous value of 4.0000 to a new value of 4.0001, the program was run a second time. Failure did not occur on the second run, and the new calculations (pa. 68) indicate filter gains of -2.0, and a "small" residue of 3.19D-14 (essentially zero). These are the correct values.

This example points out one possible method of correcting certain program failure modes, should they occur during execution.

The full terminal session is recorded below, with user input in lower case letters following each "?" .

Following the program failure example, that portion of the repeated terminal output was deleted up to the point where program execution of the second run begins.

```
reccrd on
BEGIN RECORDING OF TERMINAL SESSION
R; T=0.01,0.02 21:49:30
filedef 06 term (recfm fa blksize 133
global txtlib fortmod2 mcd2esh imsldp ncniimsl
lcad cptsysx (start

EXECUTION BEGINS...
OPTSYSX IS A COMPLETELY INTERACTIVE OPTIMAL SYSTEMS CONTROL
PROGRAM. IT WILL SOLVE NUMEROUS CONTROL PROBLEMS ON THE
FOLLOWING TYPES OF SYSTEMS CONTROL EQUATIONS:
```


XDOT = (F)*X + (G)*U + (GAM)*(W+W0)
MEASUREMENT EQUATION--
Z = (H)*X + (D)*W + V
REGULATOR PERFORMANCE INDEX--
J = 1/2 * INTEGRAL (Y *(A)*Y + U *(B)*U) DT
STATE FEEDBACK GAIN DEFINITION--
U = - (C) *X

DO YOU WISH TO CONTINUE? TYPE "YES" OR "NO".

yes

--DATA ENTRY--

ALTECUGH OPTSYSX IS SPECIFICALLY DESIGNED TO READ ALL MATRIX DATA INTERACTIVELY, SEVERAL ALTERNATE METHODS ARE AVAILABLE TO USERS:

METHOD 1--THE "F", "G", AND "GAMMA" MATRICES MAY BE READ FROM SEPARATE DATA FILES.

METHOD 2--THE "F", "G", AND "GAMMA" MATRICES MAY BE EXPLICITLY DEFINED WITHIN SUBROUTINE "SETUP".

(NOTE: IN EITHER CASE, THE USER SHOULD OBTAIN A COPY OF THE PROGRAM LISTING AND EXAMINE THE EXAMPLES CONTAINED IN S/R "SETUP".)

DO YOU WISH TO CONTINUE? TYPE "YES" OR "NO".

yes

DO YOU WISH TO INPUT THE "F", "G", AND "GAMMA" MATRICES FROM SUBROUTINE "SETUP" IAW THE METHOD DESCRIBED ON THE PREVIOUS SCREEN?

TYPE "YES" OR "NO".

no

GENERAL OPTSYSX OPTIONS:

CPTION 1 -- SYSTEM ANALYSIS WITHOUT OPEN-LOOP EIGENSYSTEM CALCULATIONS.

CPTION 2 -- SYSTEM ANALYSIS WITH OPEN-LOOP EIGENSYSTEM CALCULATIONS.

CFTION 3 -- OPEN-LOOP EIGENSYSTEM FOUND AND PROGRAM TERMINATES.

("F"--MATRIX ENTRY FOLLOWS IMMEDIATELY.)

OPTION 4 -- MODAL DISTRIBUTION MATRICES COMPUTED
WITHOUT FILTER OR REGULATOR SYNTHESIS
OF STEADY-STATE ANALYSIS.

SELECT AN OPTION: 1, 2, 3, OR 4.

?
2

DO YOU DESIRE RMS VALUES OF STATE AND CONTROL?
TYPE "YES" OR "NO".

yes

CFTSYSX LQR/CLASSICAL OPTIONS:

OPTION 1 -- OPTIMAL FILTER AND/OR REGULATOR
SYNTHESIS WITH NO EXTERNAL "C" OR "K"
MATRIX INPUT.

OPTION 2 -- OPTIMAL FILTER AND/OR REGULATOR
SYNTHESIS WITH EXTERNAL "C"
MATRIX INPUT.

OPTION 3 -- OPTIMAL FILTER AND/OR REGULATOR
SYNTHESIS WITH EXTERNAL "K"
MATRIX INPUT.

OPTION 4 -- OPTIMAL FILTER AND/OR REGULATOR
SYNTHESIS WITH EXTERNAL "C" AND "K"
MATRIX INPUT.

SELECT AN OPTION: 1, 2, 3, OR 4.

?
1

DO YOU WISH TO DETERMINE THE STEADY-STATE RESPONSE
FOR A CONSTANT DISTURBANCE?
TYPE "YES" OR "NO".

no

DO YOU WISH TO DETERMINE THE MODAL DISTRIBUTION
AND GAIN MATRICES?
TYPE "YES" OR "NO".

no

CPEN-LCOP TRANSFER FUNCTION OPTIONS:
OPTION 1 -- NO CPEN-LCOP TRANSFER FUNCTIONS COMPUTED.

CPTION 2 -- POLES, RESIDUES, AND ZEROS COMPUTED.

CPTION 3 -- ONLY POLES AND ZEROS COMPUTED.

CPTION 4 -- ONLY POLES AND RESIDUES COMPUTED.

SELECT AN CPTION: 1, 2, 3, OR 4.

?

2

NCISE TRANSFER FUNCTION OPTIONS:

CPTION 1 -- NC NOISE TRANSFER FUNCTIONS COMPUTED.

CPTION 2 -- POLES, RESIDUES, AND ZEROS COMPUTED.

CPTION 3 -- ONLY POLES AND ZEROS COMPUTED.

CPTION 4 -- ONLY POLES AND RESIDUES COMPUTED.

SELECT AN CPTION: 1, 2, 3, OR 4.

?

1

COMPENSATOR TRANSFER FUNCTION OPTIONS:

CPTION 1 -- NC COMP. TRANSFER FUNCTIONS COMPUTED.

CPTION 2 -- POLES, RESIDUES, AND ZEROS COMPUTED.

CPTION 3 -- ONLY POLES AND ZEROS COMPUTED.

CPTION 4 -- ONLY POLES AND RESIDUES COMPUTED.

(NOTE: A COMPENSATOR TRANSFER FUNCTION CAN BE
COMPUTED ONLY IF BOTH A REGULATOR
AND FILTER ARE SYNTHESIZED
AND/OR INPUT.)

SELECT AN CPTION: 1, 2, 3, OR 4.

?

1

WILL A FEED-FORWARD DISTRIBUTION MATRIX

("D" - MATRIX) BE INPUT?

TYPE "YES" OR "NO".

10

DO YOU DESIRE TO SYNTHESIZE A STABLE FILTER (OR REGULATOR)
BY DESTABILIZING THE ORIGINAL SYSTEM?

(NOTE: WORKS FOR FILTER OR REGULATOR BUT NOT FOR BOTH
IN THE SAME RUN.)

TYPE "YES" CR "NO".

NO

DO YOU DESIRE TO PRINT THE EULER-LAGRANGE EIGENSYSTEM
PRIOR TO DECOMPOSITION (FOR CHECKING THE PROGRAM)?
TYPE "YES" CR "NO".

NO

POWER SPECTRAL DENSITY (PSD) OPTION 1 :

OPTION 1 -- COMPUTE THE PSD OF THE OUTPUTS AND/OR THE
CONTROLS OF THE CONTROLLED SYSTEM WHEN FORCED BY
PROCESS AND MEASUREMENT NOISE. (NOTE: BOTH A
REGULATOR AND A FILTER MUST BE RESIDENT IN THE
PROGRAM TO USE THIS OPTION.)

OPTION 2 -- SAME AS OPTION 1 ABOVE BUT ONLY PRINT THE
RESIDUES OF EACH TRANSFER FUNCTION
USED IN THE PSD COMPUTATION.

OPTION 3 -- NOT DESIRED.

SELECT AN OPTION: 1, 2, OR 3.

?

3

DO YOU DESIRE REGULATOR SYNTHESIS ONLY?

TYPE "YES" CR "NO".

yes

ENTER THE # OF STATES (NS) OF THE SYSTEM MATRIX
("F"-MATRIX).

?

2

ENTER THE # OF CONTROLS (NC) OF THE SYSTEM MODEL
("G"-MATRIX).

?

1

ENTER THE # OF MEASUREMENTS OR OBSERVATIONS (NO)
("H"-MATRIX).

?

2

ENTER THE # OF PROCESS NOISE SOURCES (NG)

("GAMMA"-MATRIX).

?
C

FLAG/PARAMETER SETTINGS FOR THIS RUN ARE AS FOLLOWS:

IOL	IC	IR	ISS	IM	ITF1	ITF2	ITF3	IFDFW	IE	IDEBUG
1	1	C	0	0	1	0	0	0	0	0
ISET	IDSTAB	IPSD	IYU	INCRM	IREG	NS	NC	NOB	NG	
0	C	0	C	0	1	2	1	2	0	

ORDER OF SYSTEM = 2

NUMBER OF CONTROLS = 1

NUMBER OF OBSERVATIONS = 2

NUMBER OF PROCESS NOISE SOURCES = 0

ENTER THE SYSTEM MATRIX ("F"-MATRIX)

DIMENSION = # STATES (NS) X # STATES (NS)

THE ELEMENT F(1, 1) =

?
C

THE ELEMENT F(1, 2) =

?
1

THE ELEMENT F(2, 1) =

?
-1

THE ELEMENT F(2, 2) =

?
0

THE SYSTEM MATRIX ("F"-MATRIX) ...

0.0	1.00000
-1.00000	0.0

DO YOU WISH TO CHANGE THE VALUE OF ANY MATRIX ELEMENT?

TYPE "YES" OR "NO".

NO

OPEN LCCP DYNAMICS MATRIX.....F..

0.0 0.1000E+01

-0.1000E+01 0.0

OPEN LCCP EIGENVALUES.....DET(SI-F) ..

0.0 , 1.0000E+00:

OPEN LCCP RIGHT EIGENVECTOF MATRIX.....T....

0.0 -1.000000E+00

1.000000E+00 0.0

OPEN LCCP LEFT EIGENVECTOR MATRIX.....T-INV..

0.0 1.000000E+00

-1.000000E+00 0.0

ENTER THE MEASUFEMENT SCALING MATRIX ("H"-MATRIX).

DIMENSION = # OBSERVATIONS (NO) X # STATES (NS)

THE ELEMENT H(1, 1)=

?

C

THE ELEMENT H(1, 2)=

?

0

THE ELEMENT H(2, 1)=

?

C

THE ELEMENT H(2, 2)=

?

-1

THE MEASUREMENT SCALING MATRIX ("H"-MATRIX) ...

0.0 0.0

0.0 -1.00000

DO YOU WISH TO CHANGE THE VALUE OF ANY MATRIX ELEMENT?

TYPE "YES" CR "NO".

NO

MEASUREMENT SCALING MATRIX.....H..

0.0 0.0
0.0 -0.1000E+01

MCAL MEASUREMENT SCALING MATRIX...H(BAR) *T..

0.0 0.0
-1.000000D+00 0.0

ENTER THE OUTPUT MEASUREMENT COST MATRIX ("A"-MATRIX).

DIMENSION = # OBSERVATIONS (NO) X # OBSERVATIONS (NO)

THE ELEMENT A(1, 1) =

?

C

THE ELEMENT A(1, 2) =

?

0

THE ELEMENT A(2, 1) =

?

C

THE ELEMENT A(2, 2) =

?

4

THE OUTPUT MEASUREMENT COST MATRIX ("A"-MATRIX) ...

0.0 0.0
0.0 4.00000

DO YOU WISH TO CHANGE THE VALUE OF ANY MATRIX ELEMENT?

TYPE "YES" OR "NO".

NO

OUTPUT COST MATRIX.....A..

0.0 0.0
0.0 0.4000D+01

ENTER THE CONTROL DISTRIBUTION MATRIX ("G"-MATRIX).

DIMENSION = # STATES (NS) X # CONTROLS (NC)

THE ELEMENT G(1, 1) =

?

0
THE ELEMENT G(2, 1) =
?
1
THE CONTROL DISTRIBUTION MATRIX ("G"-MATRIX) ...
0.0
1.00000
DO YOU WISH TO CHANGE THE VALUE OF ANY MATRIX ELEMENT?
TYPE "YES" OR "NO".
NO
ENTER THE CONTROL COST WEIGHTING MATRIX ("B"-MATRIX)
DIMENSION = # CONTROLS (NC) X # CONTROLS (NC)
THE ELEMENT B(1, 1) =
?
1
THE CONTROL COST MATRIX.....B...
1.00000
DO YOU WISH TO CHANGE THE VALUE OF ANY MATRIX ELEMENT?
TYPE "YES" OR "NO".
NO
THE CONTROL DISTRIBUTION MATRIX.....G..
0.0
0.10000E+01
MCIAI CONTROL DISTRIBUTION MATRIX.....TI*G..
1.00000E+00
0.0
THE CONTROL COST MATRIX.....B..
0.10000E+01
OPEN LOOP TRANSFER FUNCTIONS...
TF FCF INPUT NO. 1 AND OUTPUT NO. 1:
NC FINITE ZEROS. TF GAIN = 0.0

RESIDUES AT THE POLES:

P C L E S		R E S I D U E S	
REAL (A)	IMAG (B)		
(0.0)+J(1.000000)	(0.0)	EXP (A*T)*COS (B*T)	
(0.0)+J(-1.000000)	(0.0)	EXP (A*T)*SIN (B*T)	

IF FOR INPUT NO. 1 AND OUTEUT NO. 2:

ORDERS OF NUMERATOR = 1 TF GAIN = -0.1000D+01

NUMERATOR EIGENVALUES (INCLUDING EXTRANEOUS ZERO VALUES):

(0.0)+J(0.0)
(0.0)+J(0.0)

RESIDUES AT THE POLES:

P C L E S		R E S I D U E S	
REAL (A)	IMAG (B)		
(0.0)+J(1.000000)	(-1.000000)	EXP (A*T)*COS (B*T)	
(0.0)+J(-1.000000)	(0.0)	EXP (A*T)*SIN (B*T)	

FAILURE IN HQE2 ON EIGENVALUE NO. 4

-1.962366D+00	3.464812D-03	-2.499867D+00	1.508857D+00
3.464838D-03	3.762172D-02	-1.491143D+00	2.500102D+00
-4.415041D-15	-3.208843D-13	-1.962366D+00	3.621151D-03
5.281945D-11	-1.267812D-17	3.621125D-03	3.762121D-02

EIGENSYSTEM OF OPTIMAL REGULATOR.....

EULER-LAGRANGE EQUATIONS HAVE A REAL EIGENVALUE
AT OR NEAR ZERO.

C-LCCF OPTIMAL REG. E-VALUES...DET(SI-F+G*C) ..

0.0 : 0.0 , -1.00000D+00:

C-LCCF RIGHT EIGENVECTOR MATRIX.....M....

-7.058807D-01	6.035185D-01
-7.083307D-01	1.000000D+00

CONTROL EIGENVECTOR MATRIX.....C*M..
-1.411761D+00 -1.504737D-02

C*ICCP OPT. REG. LEFT E-VECTOR MATRIX..H-INV..
-3.592082D+00 2.167888D+00
-2.544382D+00 2.535582D+00

THE OPTIMAL FEEDBACK GAIN CONTROL MATRIX...C=BINV*GT*S...
5.10951E+00 -3.0987E+00

THE MCDAL CCNTRCI GAINS.....C*T..
-3.098696D+00 -5.109451D+00

THE CLOSED LOOP DYNAMICS MATRIXF-G*C..
0.0 1.000000D+00
4.109451D+00 -3.098696D+00

ANALYSIS COMPLETE. DO YOU WANT ANOTHER RUN?
TYPE "YES" OR "NO".

yes

DO YOU WISH TO SAVE THE "F"-MATRIX FROM THE LAST
RUN TO BE USED IN THE FOLLOWING RUN?
NOTE: THE MATRIX WILL BE FEDISPLAYED AT
THE PROPER INPUT SEQUENCE INTERVAL
AND YOU WILL HAVE THE OPTION OF CHANGING
INDIVIDUAL MATRIX ELEMENTS.
TYPE "YES" OR "NO".

yes

DO YOU WISH TO SAVE THE "H"-MATRIX FROM THE LAST
RUN TO BE USED IN THE FOLLOWING RUN?
NOTE: THE MATRIX WILL BE FEDISPLAYED AT
THE PROPER INPUT SEQUENCE INTERVAL
AND YOU WILL HAVE THE OPTION OF CHANGING
INDIVIDUAL MATRIX ELEMENTS.
TYPE "YES" OR "NO".

yes

DO YOU WISH TO SAVE THE "G"-MATRIX FROM THE LAST

RUN TO BE USED IN THE FOLLOWING RUN?
NOTE: THE MATRIX WILL BE REDISPLAYED AT
THE PROPER INPUT SEQUENCE INTERVAL
AND YOU WILL HAVE THE OPTION OF CHANGING
INDIVIDUAL MATRIX ELEMENTS.

TYPE "YES" OR "NO".

yes

Author's note: Since the same program options are to
be run again, with only a change in one
of the cost matrix element values, the
terminal output was deleted up to the
point where program calculations resume
in order to avoid redundancy.

ORDER OF SYSTEM = 2
NUMBER OF CONTROLLED = 1
NUMBER OF OBSERVATIONS = 2
NUMBER OF PROCESS NOISE SOURCES = 0

THE SYSTEM MATRIX ("F"-MATRIX) ...

0.0 1.00000
-1.00000 0.0

DO YOU WISH TO CHANGE THE VALUE OF ANY MATRIX ELEMENT?
TYPE "YES" OR "NO".

no

OPEN LCCP DYNAMICS MATRIX.....F..
0.0 0.10000D+01
-0.10000D+01 0.0
OPEN LCCP EIGENVALUES.....DET(SI-F)..
0.0 , 1.00000D+00:
OPEN LCCP RIGHT EIGENVECTOR MATRIX.....T....
0.0 -1.00000D+00
1.00000D+00 0.0

CEFM LCCP LEFT EIGENVECTOR MATRIX.....T-INV..

0.0 1.000000D+00
-1.000000D+00 0.0

THE MEASUREMENT SCALING MATRIX ("H"-MATRIX) ...

0.0 0.0
0.0 -1.00000

DO YOU WISH TO CHANGE THE VALUE OF ANY MATRIX ELEMENT?
TYPE "YES" OR "NO".

NO

MEASUREMENT SCALING MATRIX.....H..

0.0 0.0
0.0 -0.1000D+01

MEDAL MEASUREMENT SCALING MATRIX...H(BAR) *T..

0.0 0.0
-1.000000D+00 0.0

ENTER THE OUTPUT MEASUREMENT COST MATRIX ("A"-MATRIX).

DIMENSION = # OBSERVATIONS (NO) X # OBSERVATIONS (NO)

THE ELEMENT A(1, 1) =

?

C

THE ELEMENT A(1, 2) =

?

0

THE ELEMENT A(2, 1) =

?

C

THE ELEMENT A(2, 2) =

?

4.0001

THE CUTPUT MEASUREMENT COST MATRIX ("A"-MATRIX) ...

0.0 0.0
0.0 4.00010

DO YOU WISH TO CHANGE THE VALUE OF ANY MATRIX ELEMENT?

TYPE "YES" OR "NO".

NO

OUTPUT COST MATRIX.....L...

0.0 0.0

0.0 0.4000E+01

THE CONTROL DISTRIBUTION MATRIX ("G"-MATRIX) ...

0.0

1.0000C

DO YOU WISH TO CHANGE THE VALUE OF ANY MATRIX ELEMENT?

TYPE "YES" OR "NO".

NO

ENTER THE CONTROL COST WEIGHTING MATRIX ("B"-MATRIX)

DIMENSION = # CONTROLS (NC) X # CONTROLS (NC)

THE ELEMENT B(1, 1) =

?

1

THE CONTROL COST MATRIX.....B...

1.0000C

DO YOU WISH TO CHANGE THE VALUE OF ANY MATRIX ELEMENT?

TYPE "YES" OR "NO".

NO

THE CONTROL DISTRIBUTION MATRIX.....G...

0.0

0.1000E+01

INITIAL CONTROL DISTRIBUTION MATRIX.....TI*G..

1.000000E+00

0.0

THE CONTROL COST MATRIX.....B..

0.1000E+01

OPEN LOOP TRANSFER FUNCTIONS...

TF FCF INPUT NO. 1 AND OUTPUT NO. 1:

NC FINITE ZEROS. TF GAIN = 0.0

RESIDUES AT THE POLES:

P C L E S		R E S I D U E S	
REAL(A)	IMAG(B)		
(0.0)+J(1.000000)	(0.0)	EXP(A*T)*COS(B*T)	
(0.0)+J(-1.000000)	(0.0)	EXP(A*T)*SIN(B*T)	

IF FCF INPUT NO. 1 AND OUTPUT NO. 2:

ORDER OF NUMERATOR = 1 TF GAIN = -0.1000D+01

NUMERATOR EIGENVALUES (INCLUDING EXTRANEOUS ZERO VALUES):

(0.0)+J(0.0)
(0.0)+J(0.0)

RESIDUES AT THE POLES:

P C L E S		R E S I D U E S	
REAL(A)	IMAG(B)		
(0.0)+J(1.000000)	(-1.000000)	EXP(A*T)*COS(B*T)	
(0.0)+J(-1.000000)	(0.0)	EXP(A*T)*SIN(B*T)	

EIGENSYSTEM OF OPTIMAL REGULATOR.....

C-LCCP OPTIMAL REG. E-VALUES...DET(SI-F+G*C)..
-1.005C1D+00:-9.95012D-01:

C-LCCP RIGHT EIGENVECTOR MATRIX.....M....
7.053368D-01 -7.088723D-01
-7.088723D-01 7.053368D-01

CONTFC1 EIGENVECTOR MATRIX.....C*M..
1.417762D+00 -1.410691D+00

C-LCCP OPT. REG. LEFT E-VECTOR MATRIX..M-INV..
-1.410691D+02 -1.417762D+02
-1.417762D+02 -1.410691D+02

THE OPTIMAL FEEDBACK GAIN CONTROL MATRIX...C=BINV*GT*S...
-3.1974D-14 -2.0000D+00

THE MODAL CONTROL GAINS.....C*T..
-2.000025D+00 3.197442D-14

THE CLOSED LOOP DYNAMICS MATRIXF-G*C..
0.0 1.000000D+00
-1.000000D+00 -2.000025D+00

ANALYSIS COMPLETE. DO YOU WANT ANOTHER RUN?
TYPE "YES" OR "NO".

rc

.....CPTSYSX IS NOW TERMINATED.....

R: T=0.63/2.60 23:33:07

record off

END RECORDING OF TERMINAL SESSION

V. CONCLUSIONS AND RECOMMENDATIONS

A. CONCLUSIONS

Although originally developed for the quadratic synthesis of controllers for rotary-wing VTOL aircraft, the extensive modifications and enhancements of Hall's original work, coupled with its efficient and accurate eigensystem solution routine, represent a powerful tool in the design of optimally controlled systems.

In its present interactive form, OPTSYSX has been transferred from the arena of high-level applied mathematics and numerical analysis to the level of control system engineers and students. It now represents an even more powerful educational tool, able to rapidly and effectively unlock many misunderstood linear systems mathematical relationships.

As an ultimate evaluation of the computational abilities of OPTSYSX, the program was tested using an 82 X 82 matrix of aircraft longitudinal motion equations for the VX-29 experimental Fighter aircraft derivative, provided by NASA-Fairchild.

For a system of equations of this magnitude, all program arrays were re-dimensioned (as shown in Appendix A), and a 2-Megabyte virtual machine size was required. This system was run through the Modal Analysis option of OPTSYSX, requiring less than 90 seconds to load the system and complete all open-loop and modal analysis calculations!

Program results exhibited perfect eigenvalue correlation with those obtained from the John Edwards Control Program. Additionally, OPTSYSX provided complete longitudinal modal analysis, previously unavailable on a system of this size.

It is hoped that the use of this interactive program version will be encouraged; and that its expanded abilities will stimulate both interest in and research on basic systems control problems, as well as more advanced designs.

E. RECOMMENDATIONS

Based on the results of this thesis, four areas emerged as possibilities for further research and study:

1. Program Availability

The use of CPTSYSX and similar design programs should be encouraged in all undergraduate and graduate level courses involved in the analysis and design of control systems. Toward this end, it is recommended that CPTSYSX be placed in the non-IMSL library of subroutines, making it easily available to all potential users.

2. Computer Graphics

The addition of graphical plotting routines to the program in the time and frequency domain would make CPTSYSX an even more powerful tool in the design of many optimally controlled systems.

3. Further Modifications

The present version of the program should be modified to include the CPTSYS 5 derivative input term improvements of Liu [Ref. 3], and program sequencing during optimal filter synthesis should be examined. Various test runs indicate an area of conflict in that the program appears to require the design of an optimal regulator prior to performing any filter calculations.

4. Program Application

CPTSYSX offers attractive possibilities in the area of microcomputer implementation.

APPENDIX A
OPTSYSX PROGRAM LISTING

```

C***** OPTSYSX
C***** BY JOHN G. HODEN
C***** THIS PROGRAM IS A COMPLETELY INTERACTIVE
C***** OPTIMAL SYSTEMS CONTROL DESIGN/SYNTHESIS
C***** PROGRAM CAPABLE OF HANDLING VERY LARGE (80X80) +
C***** MULTIVARIABLE SYSTEMS OF LINEAR EQUATIONS.
C***** VERSION 1.8 11 MAR 1984
C***** IMPLICIT REAL*8(A-H,C-Z)
C----- INTEGER IANS, ICL, IQ, IR, ISS, IM, ITF1, ITF2, ITF3, IPDFW, IE, IDEBUG, ISET,
C----- IIPSD, IYU, INORM, NS, NC, NOB, NG, IBEG, IDSTAE, IRET, NROW, NCOL, ISAF, ISAH, I
C----- 2SAG, IGM
C----- LARGE QRDER SYSTEM (82 X 82) DIMENSIONS.
C----- DIMENSION ACL(82,82), B(41,41), EA(82,82), CI(82), CR(82), CO(82,82),
C----- *CWI(82), CWR(82), FEGC(41,82), FEGE(82,41), G(82,82), GM(82,82),
C----- *FBO(82,82), RC(41,41), SC(82,82), WR(164), WI(164), W11(82,82),
C----- *W21(82,82), X(164,164), GN(82,82), HO(41,82), D1(164), D2(164),
C----- *RM(164,164), Q(41,41), GAM(82,41), WNORM(82,82), WNORMI(82,82),
C----- *DESTAB(82), AA(82,82), BM(82,41), CM(41,82), D(41,41), DSTORE(82,82),
C----- *JCF(164), RES(164), AY(82,82), BE(164), CC(164), CP(82), GW(164,41),
C----- *GV(164,41), HY(41,164), HU(41,164), PRTT(16,16), DUM(82,85)
C----- STANDA RD PROGRAM DIMENSIONS.
C----- DIMENSION ACL(32,32), B(32,32), EA(32,32), CI(32), CR(32), CO(32,32), C
C----- 1I(32), CWR(32), FEGC(32,32), FEGE(32,32), G(32,32), GM(32,32), PRO(32,32),
C----- 2, RC(32,32), SC(32,32), WR(64), WI(64), W11(32,32), W21(32,32), X(64,64),
C----- 3, GN(32,32), HO(32,32), D1(64), D2(64), RM(64,64), Q(32,32), GAM(32,32),
C----- 4, NCBM(32,32), WNORM(32,32), DESTAB(32), AA(32,32), BM(32,32), CM(32,32),
C----- 5, D(32,32), DSTORE(32,32), JCF(64), RES(64), AY(32,32), BB(64), CC(64),
C----- 6(32,64), GW(64,64), GV(64,64), HY(64,64), HU(64,64), PRTT(16,16), DUM(32,32
C----- 7)
C----- EQUIVALENCE (W11(1,1), GW(1,1)), (W11(1,1), GV(1,1)), (W21(1,1), HY(1
C----- 1,1)), (W21(1,1), HU(1,1))
C----- CCOMMON /PROG/ IOL, IQ, IR, ISS, IM, ITF1, ITF2, ITF3, IPDFW, IE, IDSTAE, IDEB
C----- 1UG, ISET, IBEG, IIPSD, IYU, INORM
C----- DATA IY/'Y'/, IZ/'N'/
C----- SUPPRESS INDIVIDUAL UNDERFLOW, OVERFLOW, DIVIDE CHECK, AND DECIMAL =
C----- CONVENT ERRCR MESSAGES; PROVIDE SUMMARY OF ERRORS ONLY.
C----- CALL ERRSET (207,256,-1,1,1,209)
C----- CALL ERRSET (215,256,-1,1)
C----- INITIALIZE FLAGS.
C----- ISAF=0
C----- ISAG=0
C----- ISAH=0
C----- IGAM=0
10  CCNTINUE
C----- IRET=0
C----- ICL=0
C----- IC=0
C----- IB=0
C----- ISS=0
C----- IM=0
C----- ITF1=0
C----- ITF2=0
C----- ITF3=0
C----- IPDFW=0
C----- IE=0
C----- IDSTAB=0

```



```

I DEBUG=0
I SET=0
I PSD=0
I YU=0
I NORM=0
I REG=0
N S=0
N C=0
N CB=0
N G=0
C-----SCRN1-----
20  CALL PRTCMS ('CLRSCE1 ')
  WRITE (5,890)
  CALL RDCHAR (IANS)
  IF ((IANS.NE.IY).AND.(IANS.NE.IZ)) GO TO 30
  GO TO 40
30  WRITE (5,880)
  GO TO 20
40  CONTINUE
  IF (IANS.EQ.IZ) GC TO 560
C-----SCRN2-----
50  CALL PRTCMS ('CLRSCE1 ')
  WRITE (5,900)
  CALL RDCHAR (IANS)
  IF ((IANS.NE.IY).AND.(IANS.NE.IZ)) GC TO 60
  GO TO 70
60  WRITE (5,880)
  GO TO 50
70  CONTINUE
  IF (IANS.EQ.IZ) GC TO 560
C-----ISET-----
80  CALL PRTCMS ('CLRSCE1 ')
  WRITE (5,910)
  CALL RDCHAR (IANS)
  IF ((IANS.NE.IY).AND.(IANS.NE.IZ)) GC TO 90
  GO TO 100
90  WRITE (5,880)
  GO TO 80
100 CONTINUE
  IF (IANS.EQ.IY) ISET=1
C-----IOL-----
  CALL PRTCMS ('CLRSCE1 ')
  WRITE (5,570)
  CALL RDINT (IANS)
  ICL=IANS-1
  IF (IOL.EQ.2) GO TO 350
C-----IQ-----
110  CALL PRTCMS ('CLRSCE1 ')
  WRITE (5,580)
  CALL RDCHAR (IANS)
  IF ((IANS.NE.IY).AND.(IANS.NE.IZ)) GO TO 120
  GO TO 130
120  WRITE (5,880)
  GO TO 110
130  CONTINUE
  IF (IANS.EQ.IY) IC=1
  IF (IANS.EQ.IZ) IC=0
  IF (IOL.EQ.3) GO TO 200
C-----IR-----
  CALL PRTCMS ('CLRSCE1 ')
  WRITE (5,590)
  CALL RDINT (IANS)
  IR=IANS-1
C-----ISS-----
140  CALL PRTCMS ('CLRSCE1 ')
  WRITE (5,600)
  CALL RDCHAR (IANS)
  IF ((IANS.NE.IY).AND.(IANS.NE.IZ)) GC TO 150
  GO TO 160
150  WRITE (5,880)
  GO TO 140
160  CONTINUE
  IF (IANS.EQ.IY) ISS=1
  IF (IANS.EQ.IZ) ISS=0
C-----IM-----
170  WRITE (5,610)

```



```

CALL RDCHAR (IANS)
IF ((IANS.NE.IY).AND.(IANS.NE.IZ)) GO TO 180
180 GO TO 190
190 CONTINUE
IF (IANS.EQ.IY) IM=1
IF (IANS.EQ.IZ) IM=0
200 CONTINUE
IF (IOL.EQ.3) IM=1
C-----ITF1-----
CALL FRTCMS ('CLRSCEN ')
WRITE (5,680)
CALL RDINT (IANS)
ITF1=IANS-1
C IF (IOL.EQ.3) GO TO 240
C-----ITF2-----
CALL FRTCMS ('CLRSCEN ')
WRITE (5,630)
CALL RDINT (IANS)
ITF2=IANS-1
C IF (IOL.EQ.3) GO TO 240
C-----ITF3-----
CALL FRTCMS ('CLRSCEN ')
WRITE (5,640)
CALL RDINT (IANS)
ITF3=IANS-1
C-----IFDFW-----
CALL FRTCMS ('CLRSCEN ')
210 WRITE (5,650)
CALL RDCHAR (IANS)
IF ((IANS.NE.IY).AND.(IANS.NE.IZ)) GC TO 220
GO TO 230
220 WRITE (5,880)
GO TO 210
230 CONTINUE
IF (IANS.EQ.IY) IFDFW=1
IF (IANS.EQ.IZ) IFDFW=0
C-----IE-----
CALL FRTCMS ('CLRSCEN ')
WRITE (5,660)
CALL RDREAL (ANSR)
IE=IDINT (ANSR)
IF (IOL.EQ.3) GO TO 300
C-----IDSTAB-----
CALL FRTCMS ('CLRSCEN ')
240 WRITE (5,670)
CALL RDCHAR (IANS)
IF ((IANS.NE.IY).AND.(IANS.NE.IZ)) GC TO 250
GO TO 260
250 WRITE (5,880)
GO TO 240
260 CONTINUE
IF (IANS.EQ.IY) IDSTAB=1
IF (IANS.EQ.IZ) IDSTAB=0
C-----IDEBUG-----
270 WRITE (5,680)
CALL RDCHAR (IANS)
IF ((IANS.NE.IY).AND.(IANS.NE.IZ)) GO TO 280
GO TO 290
280 WRITE (5,880)
GO TO 270
290 CONTINUE
IF (IANS.EQ.IY) IDEBUG=1
IF (IANS.EQ.IZ) IDEBUG=0
300 CONTINUE
C-----IPSD-----
CALL FRTCMS ('CLRSCEN ')
WRITE (5,690)
CALL RDINT (IANS)
IPSD=IANS
IF (IPSD.EQ.3) IPSD=0
IF (IPSD.EQ.0) GO TO 310
C-----IYU-----
CALL FRTCMS ('CLRSCEN ')
WRITE (5,700)

```



```

CALL RDINT (IANS)
IYU=IANS-1
C-----INORM-----
CALL PRTCMS ('CLRSCE1 ')
WRITE (5,820)
CALL RDREAL (ANSR)
INORM=IDINT(ANSR)
310 IF (IOL.EQ.3) GO TO 350
C-----IREG-----
CALL PRTCMS ('CLRSCE1 ')
320 WRITE (5,710)
CALL RDCHAR (IANS)
IF ((IANS.NE.IY).AND.(IANS.NE.IZ)) GO TO 330
GO TO 340
330 WRITE (5,880)
GO TO 320
340 CCNTINUE
IF (IANS.EQ.IY) IREG=1
IF (IANS.EQ.IZ) IREG=0
C-----NS-----
350 CALL PRTCMS ('CLRSCE1 ')
WRITE (5,720)
CALL RDREAL (ANSR)
NS=IDINT(ANSR)
IF (IOL.EQ.2) GO TO 360
C-----NC-----
WRITE (5,730)
CALL RDREAL (ANSR)
NC=IDINT(ANSR)
C-----NOB-----
WRITE (5,740)
CALL RDREAL (ANSR)
NOB=IDINT(ANSR)
C-----NG-----
WRITE (5,750)
CALL RDREAL (ANSR)
NG=IDINT(ANSR)
360 CCNTINUE
C-----FLAG SETTINGS-----
CALL PRTCMS ('CLRSCE1 ')
WRITE (5,760)
WRITE (5,770)
WRITE (5,780) IOL,IQ,IR,ISS,IP,ITF1,ITF2,ITF3,IPDFW,IE,IDEBUG,ISET
1,ILSTAB
WRITE (5,790)
WRITE (5,800) IPSE,IYU,INORM,IREG,NS,NC,NOB,NG
WRITE (5,810) NS,NC,NOB,NG
C-----BEGIN CALCULATIONS-----
N2=2*NS
CALL INNER (NS,NC,NOB,NG,N2,ACL,B,BA,CI,CR,CO,CWI,CWR,D,FBGC,FBGE,
1G,GAM,GM,GN,HC,D1,D2,PRO,EM,EC,SC,AR,NI,W11,W21,X,WNCRM,WNOFMI,D
2ESTAB,AA,BM,CM,JCF,RES,AY,BB,CC,CP,GW,GV,AY,HU,DSTORE,ISAF,ISAH,IS
3AG,IGAM,IRET,FRRT,NRCW,NCCL)
C-----IRET-----
370 WRITE (5,830)
CALL RDCHAR (IANS)
IF ((IANS.NE.IY).AND.(IANS.NE.IZ)) GO TO 380
GO TO 390
380 WRITE (5,880)
GO TO 370
390 CCNTINUE
IF (IANS.EQ.IY) GO TO 400
IF (IANS.EQ.IZ) GO TO 560
C-----ISAF-----
400 CCNTINUE
IF (IRET.EQ.1) GO TO 10
IF (ISET.EQ.1) GO TO 10
CALL PRTCMS ('CLRSCE1 ')
410 WRITE (5,840)
CALL RDCHAR (IANS)
IF ((IANS.NE.IY).AND.(IANS.NE.IZ)) GO TO 420
GO TO 430
420 WRITE (5,880)
GO TO 410
430 CCNTINUE
IF (IANS.EQ.IY) ISAF=1

```



```

C----- IF (IANS.EQ.IZ) ISAF=0 -----ISAH-----
440 IF (NOB.EQ.0) GO TO 470
CALL FRCTMS ('CLRSCHN ')
WRITE (5,850)
CALL RDCEAR (IANS)
IF ((IANS.NE.IY).AND.(IANS.NE.IZ)) GC TO 450
450 GO TO 460
460 WRITE (5,880)
GO TO 440
CONTINUE
IF (IANS.EQ.IY) ISAH=1
IF (IANS.EQ.IZ) ISAH=0
470 CCNTINUE
C----- IF (NC.EQ.0) GC TO 510 -----ISAG-----
480 WRITE (5,860)
CALL RDCEAR (IANS)
IF ((IANS.NE.IY).AND.(IANS.NE.IZ)) GC TO 490
490 GO TO 500
500 WRITE (5,880)
GO TO 480
CONTINUE
IF (IANS.EQ.IY) ISAG=1
IF (IANS.EQ.IZ) ISAG=0
510 CCNTINUE
C----- IF (NG.EQ.0) GC TO 550 -----IGAM-----
520 WRITE (5,870)
CALL RDCEAR (IANS)
IF ((IANS.NE.IY).AND.(IANS.NE.IZ)) GC TO 530
530 GO TO 540
540 WRITE (5,880)
GO TO 520
CONTINUE
IF (IANS.EQ.IY) IGAM=1
IF (IANS.EQ.IZ) IGAM=0
550 CCNTINUE
GO TO 10
C----- WRITE (5,920) -----TERMINATE-----
560 STCP
C----- FORMAT (25X,24HGENERAL OPTSYSX CPTIONS: //,10X,35HOPTION 1 -- SYST
570 12M ANALYSIS WITHOUT./,22X,35HCPEN-LOCP EIGENSYSTEM CALCULATIONS./
2//,10X,42HOPTION 2 -- SYSTEM ANALYSIS WITH OPEN-LOOP /,22X,25HEIGEN
3SYSTEM CALCULATIONS.//,10X,39HCPTION 3 -- OPEN-LOOP EIGENSYSTEM F
40UND/,22X,23HAND PRGGRAM TERMINATES./,22X,39H "F"-MATRIX ENTRY F
5OLLOW//,10X,48HCPTION 4 -- MODAL DISTRIBUTION MATR
6ICES COMPUTED./,22X,37HWITHCUT FILTER AND REGULATOR SYNTHESIS./,22X
7,25HOR STEADY-STATE ANALYSIS./,15X,30HSELECT AN OPTION: 1,2,3,0
8R 4.)
580 FORMAT (/,5X,46HDO YOU DESIRE RMS VALUES OF STATE AND CONTRCL?,//,
1,10X,19HTYPE "YES" CR "NO".)
590 FORMAT (/,20X,30HCPTSYSX LOR/CLASSICAL OPTIONS: //,10X,43HOPTION 1
1 -- OPTIMAL FILTER AND/OR REGULATOR/,,22X,37HSYNTHESIS WITH NC EXT
2 ETERNAL "C" OR "K",/,,22X,13HMATRIX INPUT/,,10X,43HOPTION 2 -- OPTI
3MAL FILTER AND/OR REGULATOR/,,22X,27HSYNTHESIS WITH EXTERNAL "C"/
4,,22X,13HMATRIX INPUT/,,10X,43HCPTION 3 -- OPTIMAL FILTER AND/OR
5REGULATOR/,,22X,27HSYNTHESIS WITH EXTERNAL "K"/,,22X,13HMATRIX INP
6UT/,,10X,43HCPTION 4 -- OPTIMAL FILTER AND/OR REGULATOR/,,22X,35
7HSYNTHESIS WITH EXTERNAL "C" AND "K"/,,22X,13HMATRIX INPUT/,,10X
8,,32HSELECT AN OPTION: 1, 2, 3, OR 4.)
600 FORMAT (5X,5HDO YOU WISH TO DETERMINE THE STEADY-STATE RESPON
12,/,3X,24HFOR A CCNSTANT CISTURBANCE?,/,10X,19HTYPE "YES" CR "NO"
2.)
610 FORMAT (5X,47HDO YOU WISH TO DETERMINE THE MODAL DISTRIBUTION/,,8X
1,18HAND GAIN MATRICES?,/,10X,19HTYPE "YES" OR "NO".)
620 FORMAT (/,5X,36HCPEN-LOOP TRANSFER FUNCTION OPTIONS: //,10X,53HCP
1TION 1 -- NO OPEN-LOOP TRANSFER FUNCTIONS COMPUTED.//,10X,48HCPII
2ON 2 -- POLES, RESIDUES, AND ZEROS COMPUTED.//,10X,42HOPTION 3 --
3 ONLY POLES AND ZEROS COMPUTED.//,10X,45HCPTION 4 -- ONLY POLES A
4ND RESIDUES COMPUTED//,10X,32HSELECT AN OPTION: 1, 2, 3, OR 4.)
630 FORMAT (/,5X,32HNOISE TRANSFER FUNCTION OPTIONS: //,10X,49HOPTION

```



```

1 1 -- NO NOISE TRANSFER FUNCTIONS COMPUTED.//,10X,48HOPTION 2 --
2 POLES, RESIDUES, AND ZEROS COMPUTED.//,10X,42HOPTION 3 -- ONLY PO
3 LES AND ZEROS COMPUTED.//,10X,45HOPTION 4 -- ONLY POLES AND RESID
4 UES COMPUTED.//,10X,32HSELECT AN OPTION: 1, 2, 3, OR 4.)
640 FORMAT //,5X,38HCOMPENSATOR TRANSFER FUNCTION OPTIONS: //,10X,49H
1OPTION 1 -- NO CCMP. TRANSFER FUNCTIONS COMPUTED.//,10X,48HOPTION
2 2 -- POLES, RESIDUES, AND ZEROS COMPUTED.//,10X,42HOPTION 3 -- O
3 ONLY POLES AND ZEROS COMPUTED.//,10X,45HOPTION 4 -- ONLY POLES AND
4 RESIDUES COMPUTED.//,10X,45HNOTE: A COMPENSATOR TRANSFER FUNCTI
5 SON CAN BE //,22X,33HCOMPUTED ONLY IF BOTH A REGULATOR, //,22X,26H AND
6 FILTER ARE SYNTHESIZED.//,22X,14HAND/CR INPUT.//,10X,32HSELECT AN
7 OPTION: 1, 2, 3, OR 4.)
650 FORMAT //,5X,39H WILL A FEED-FORWARD DISTRIBUTION MATRIX.//,5X,25H
1 "D" - MATRIX BE INPUT? //,15X,19HTYPE "YES" OR "NO".)
660 FORMAT //,5X,63H THIS OPTION DETERMINES THE CRITERIA FOR DECIDING
1 WHEN A MARKOV //,8X,58HPARAMETER IS ZERO-THE MARKOV PARAMETER INDIC
2ATES THE ORDER //,8X,54HOF THE NUMERATOR POLYNOMIAL OF EACH TRANSFE
3R FUNCTION.//,8X,52HALL "N" ZEROS OF THIS POLYNOMIAL ARE PRINTED
4 OUT AND //,8X,52HTHIS TEST TELLS HOW MANY EXTRA ROOTS EXIST AT Z =
50. //,8X,41HLESS THAN 10.0 -- IE IS CONSIDERED ZERO.//,8X,47H THE
6 DEFAULT VALUE OF THIS PARAMETER IE IS //,8X,28HIN OTHER WORDS
7, IE = 1.0E-6.//,10X,66HIF YOU DESIRE A DIFFERENT MARKOV CRITERIA
8, TYPE THE INTEGER VALUE.//,10X,48HIF YOU DESIRE THE DEFAULT VALU
9E, TYPE "0" ZERO.)
670 FORMAT //,5X,61HDO YOU DESIRE TO SYNTHESIZE A STABLE FILTER OR R
1 EGULATOR //,8X,34HDESTABILIZING THE ORIGINAL SYSTEM?//,12X,52H
2 NOTE: WORKS FOR FILTER OR REGULATOR BUT NOT FOR BOTH.//,20X,17HIN T
3 HE SAME RUN //,10X,19HTYPE "YES" OR "NO".)
680 FORMAT //,5X,53HDO YOU DESIRE TO PRINT THE EULER-LAGRANGE EIGENSTY
1 M //,5X,50HPRICE TO DECOMPOSITION FOR CHECKING THE PROGRAM?//,10
2 X,19HTYPE "YES" OR "NO".)
690 PCBMAT //,5X,39HPOWER SPECTRAL DENSITY PSD OPTION 1: //,10X,53
1 HOPTION 1 -- COMPUTE THE PSD OF THE CUTPUTS AND/OR THE //,22X,48HCO
2 NTROLS OF THE CONTROLLED SYSTEM WHEN FORCED BY //,22X,45HPROCESS AN
3 D MEASUREMENT NOISE. NOTE: ECTH A //,22X,29HREGULATOR AND A FILTE
4 R MUST BE RESIDENT IN THE //,22X,29HPROGRAM TO USE THIS OPTION.//,
5,10X,53HOPTION 2 -- SAME AS OPTION 1 ABOVE BUT ONLY PRINT THE //,22
6 X,34HRESIDUES OF EACH TRANSFER FUNCTION //,22X,28HUSED IN THE PSD C
7 OMPUTATION.//,10X,24HOPTION 3 -- NOT DESIRED.//,10X,29HSELECT A
8 N OPTION: 1, 2, OR 3.)
700 FORMAT //,5X,39HPOWER SPECTRAL DENSITY PSD OPTION 2: //,10X,35
1 HOPTION 1 -- PSD CUTPUT NOT DESIRED.//,10X,38HOPTION 2 -- COMPUTE
2 ONLY OUTPUT PSD'S //,10X,39HOPTION 3 -- COMPUTE ONLY CONTROL PSD
3'S //,10X,32HSELECT AN OPTION: 1, 2, 3, OR 4.)
710 FORMAT //,5X,39HDO YOU DESIRE REGULATOR SYNTHESIS ONLY?//,10X,19
1 HTYPE "YES" OR "NO".)
720 FORMAT //,5X,47HENTER THE # OF STATES NS OF THE SYSTEM MATRIX.//,
15X,13H "F" MATRIX.)
730 FORMAT //,5X,56HENTER THE # OF CONTROLS NC OF THE CONTROL SYSTEM
1 MODEL //,5X,13H "G" MATRIX.)
740 FORMAT //,5X,54HENTER THE # OF MEASUREMENTS OR OBSERVATIONS NO O
1 F THE //,5X,13H "H" MATRIX.)
750 FORMAT //,5X,48HENTER THE # OF PROCESS NOISE SOURCES NG OF THE //,
1 5X,17H "GAMMA" MATRIX.)
760 FORMAT (5X,52HFLAG/PARAMETER SETTINGS FOR THIS RUN ARE AS FOLLOWS:
1 /)
770 FORMAT (1X,3H10,2X,2H10,2X,2H1R,2X,3H1SS,2X,2H1M,2X,4H1TF1,2X,4H1
1 TF2,2X,4H1TF3,2X,5H1FDPW,2X,2E1E,2X,6H1DEBUG,2X,4H1S1T,2X,6H1CSTAB
2 /)
780 PCBMAT (1X,12,3X,12,3X,12,2X,12,3X,12,3X,12,4X,12,4X,12,4X,1
12,3X,12,6X,12,5X,12,1 /)
790 PCBMAT (1X,4H1FSD,2X,3H1YU,2X,5H1NORM,2X,4H1REG,2X,2HNS,2X,2HNC,2X
1 3HNOB,2X,2HNG /)
800 FORMAT (2X,12,3X,12,4X,12,5X,12,3X,12,2X,12,3X,12,2X,12,1 /)
810 FORMAT (2X,17HCRDES OF SYSTEM = 13, //,2X,20HNUMBER OF CONTROLS = 1
13, //,2X,24HNUMBER OF OBSERVATIONS = 13, //,2X,33HNUMBER OF PROCESS
2NCISE SOURCES = 13, // /)
820 FORMAT (5X,53HDETERMINE THE NORMALIZATION PARAMETER INORM FOR TH
1 E, //,5X,55HPOWER SPECTRAL DENSITY PSD OPTION YOU HAVE PREVIOUSLY
2 //,5X,52HCHOSEN. TWO PSD NORMALIZATION METHODS ARE AVAILABLE: //,10
3 X,54HMETHOD 1 -- PSD IS NORMALIZED BY THE I-NORM PROCESS //,21X,49H NOTE: "Q" IS AN OPTIMAL
4 25HNOISE MINUS "Q" INORM, INORM //,21X,49H INORM = 0, 1, 2, ...
5 STATE WEIGHTING MATRIX. //,21X,34HIN THIS METHOD, INORM = 0, 1, 2, ...
6 NG, //,10X,63HMETHOD 2 -- PSD IS NORMALIZED BY THE INORM - NG, INORM - NG //,21X
7 MEASUREMENT //,21X,39HNOISE MINUS "R" INCRM - NG, INORM - NG //,21X

```


8,51H NOTE: "R" IS AN OPTIMAL CONTROL WEIGHTING MATRIX. //,21X,44H
 IN THIS METHOD, INCRK = NG + 1... NG + NOE //,10X,51H SELECT AN IN
 STEGER FROM 0 - 16 REPRESENTING YOUR PSD, //,15X,27H NORMALIZATION REQ
 UIREMENTS...//,10X,53H IF PSD NORMALIZATION IS NOT DESIRED ENTER "0"
 5 ZERO

830 FORMAT (5X,43H ANALYSIS COMPLETE. DO YOU WANT ANOTHER RUN?, //,15X,19
 1HTYPE "YES" OR "NO".)
 840 FORMAT (//,5X,48ED0 YOU WISH TO SAVE THE "F"-MATRIX FROM THE LAST
 1//,5X,36HRUN TO BE USED IN THE FOLLOWING RUN? //,5X,39H NOTE: THE M
 2ATRIX WILL BE REDISPLAYED AT //,5X,34H THE PROPER INPUT SEQUENCE INT
 3EVAL, //,5X,40H AND YOU WILL HAVE THE OPTION OF CHANGING, //,5X,27H IND
 4IVIDUAL MATRIX ELEMENTS. //,15X,19HTYPE "YES" OR "NO".)
 850 FORMAT (//,5X,48ED0 YOU WISH TO SAVE THE "H"-MATRIX FROM THE LAST
 1//,5X,36HRUN TO BE USED IN THE FOLLOWING RUN? //,5X,39H NOTE: THE M
 2ATRIX WILL BE REDISPLAYED AT //,5X,34H THE PROPER INPUT SEQUENCE INT
 3EVAL, //,5X,40H AND YOU WILL HAVE THE OPTION OF CHANGING, //,5X,27H IND
 4IVIDUAL MATRIX ELEMENTS. //,15X,19HTYPE "YES" OR "NO".)
 860 FORMAT (//,5X,48ED0 YOU WISH TO SAVE THE "G"-MATRIX FROM THE LAST
 1//,5X,36HRUN TO BE USED IN THE FOLLOWING RUN? //,5X,39H NOTE: THE M
 2ATRIX WILL BE REDISPLAYED AT //,5X,34H THE PROPER INPUT SEQUENCE INT
 3EVAL, //,5X,40H AND YOU WILL HAVE THE OPTION OF CHANGING, //,5X,27H IND
 4IVIDUAL MATRIX ELEMENTS. //,15X,19HTYPE "YES" OR "NO".)
 870 FORMAT (//,5X,52ED0 YOU WISH TO SAVE THE "GAMMA"-MATRIX FROM THE
 1LAST //,5X,36HRUN TO BE USED IN THE FOLLOWING RUN? //,5X,39H NOTE: T
 2HE MATRIX WILL BE REDISPLAYED AT //,5X,34H THE PROPER INPUT SEQUENCE
 3 INTERVAL, //,5X,40H AND YOU WILL HAVE THE OPTION OF CHANGING, //,5X,27
 4H INDIVIDUAL MATRIX ELEMENTS. //,15X,19HTYPE "YES" OR "NO".)
 880 FORMAT (5X,51H WARNING: IMPROVES DATA ENTRY! ENTER "YES" OR "NO".)
 890 FORMAT (5X,59H OPTSYSX IS A COMPLETELY INTERACTIVE OPTIMAL SYSTEMS
 1 CONTROL, //,8X,55H PROGRAM. IT WILL SOLVE NUMEROUS CONTROL PROBLEMS O
 2N THE //,3X,45H FOLLOWING TYPES OF SYSTEMS CONTROL EQUATIONS: //,15X
 3,35H DODT = P * X + G * U + GAM * (W + W0), //,20X,22H MEASUREMENT EQUA
 4TION-- //,15X,21H2 = H * X + D * W + V, //,20X,29H REGULATOR PERFORM
 5ANCE INDEX-- //,15X,42H J = 1/2 * INT EGRAL (Y * A * Y + U * B * U) DT,
 6//,20X,32H STATE FEEDBACK GAIN DEFINITION-- //,25X,10H U = - C * X, //
 7//,15X,45H DO YOU WISH TO CONTINUE? TYPE "YES" OR "NO".)
 900 FORMAT (25X, 14H--DATA ENTRY-- //,5X, 49H ALTHOUGH OPTSYSX IS SPECIFI
 1CALLY DESIGNED TO READ //,5X,48H ALL MATRIX DATA INTERACTIVELY, SEVE
 2AL ALTERNATE //,5X,31H METHODS ARE AVAILABLE TO USERS: //,10X,43H ME
 3THOD 1--THE "F", "G" AND "GAMMA" MATRICES, //,13X,37H MAY BE READ PRO
 4M SEPARATE DATA FILES. //,10X,50H METHOD 2--THE "F", "G", AND "GAMMA"
 5" MATRICES MAY BE //,13X,45H EXPLICITLY DEFINED WITHIN SUBROUTINE "S
 6ETUP". //,10X,52H NOTE: IN EITHER CASE, THE USER SHOULD OBTAIN A C
 7OPY //,15X,34H OF THE PROGRAM LISTING AND EXAMINE //,17X,39H THE EXAMP
 8LES CONTAINED IN S/R "SETUP". //,10X,45H DO YOU WISH TO CONTINUE?
 9 TYPE "YES" OR "NO".)
 910 FORMAT (//,5X,46ED0 YOU WISH TO INPUT THE "F", "G", AND "GAMMA", //
 11CX,40H MATRICES FROM SUBROUTINE "SETUP" IN THE //,10X,40H METHOD DE
 2SCRIBED ON THE PREVIOUS SCREEN? //,15X,19HTYPE "YES" OR "NO".)
 920 FORMAT (//,41H.....OPTSYSX IS NOW TERMINATED.....//)
 ENC


```

C=====
C SUBROUTINE SETUP (EA,G,GAM,NS,NC,NG)
C=====
C IMPLICIT REAL*8 (A-H,O-Z)
C DIMENSION BA(NS,NS),G(NS,NC),GAM(NS,NG),DUM(82,85)
C COMMON /FBORG/ IOL,IO,IR,ISS,ITF1,ITF2,ITF3,IPDFW,IE,LDSTAB,IDES
C 1UG,ISET,IREG,IESD,IYU,INORM
C-----
C FILE DEFINITIONS
C-----
C CALL FBTcms ('FILEDEF ','03      ','DISK  ','X29A82  ',
C 1      'DATA      ')
C-----
C THIS IS AN EXAMPLE OF AN 82 X 85 DATA FILE X29A82 DATA A1 READ FROM
C A USER'S DISK AND CONVERTED (FROM A "DUMMY" ARRAY NAMED 'DUM') TO A
C SYMMETRIC ARRAY. THE FCRMAT STATEMENT MUST MATCH YOUR DISK DATA
C FCBMAT OR THE PROGRAM WILL FAIL! NOTE: ALL PROGRAM DIMENSIONS
C MUST BE ENLARGED ACCORDINGLY FOR A SYSTEM OF THIS SIZE.
C-----
C READ (3,50) ((DUM(I,J),J=1,85),I=1,NS)
C DO 20 I=1,NS
C DO 10 J=1,NS
C 10  BA(I,J)=DUM(I,J)
C 20  CONTINUE
C-----
C THESE ARE EXAMPLES OF SEVERAL POSSIBLE METHODS OF ARRAY GENERATION
C WITHIN SUBROUTINE SETUP. THE "GAM" ARRAY WAS SET TO ZERO SINCE NO
C "NCISE" WAS PRESENT, AND THE NON-ZERO ELEMENTS OF THE "G" ARRAY WERE
C EXPLICITLY DEFINED. THEY COULD ALSO BE READ FROM FILES AS ABOVE.
C-----
C 30  DC 40 I=1,NS
C 30  DO 30 J=1,NC
C 30  GAM(I,J)=0.0D+00
C 30  G(I,J)=0.0D+00
C 30  G(82,1)=0.1000E+01
C 30  CONTINUE
C 40  CONTINUE
C 40  RETURN
C-----
C 50  FORMAT (5(212.4))
C 50  END

```



```

C=====
C      SUBROUTINE CHECK (EPS,NC,NG,NCIRET)
C      CHECKS THE CONSISTENCY OF REQUESTED OPTIONS.
C=====
C      DOUBLE PRECISION EPS
C      COMMON /EROG/ IOL,IQ,IR,ISS,IM,ITF1,ITF2,ITF3,IFDFW,I2,IDSTAB,IDE
C      IUG,ISET,IREG,IPSD,IYU,INORM
C      --SET MODAL ANALYSIS WHEN OL EIGENSY OR OL TF REQUESTED-----
C      IF (IM .EQ. 1 .AND. IOL .EQ. 0) IOL=1
C      IF (IOL .EQ. 3 .OR. ITF1 .NE. 0) IM=1
C      --CHECK TO SEE IF H MATRIX INPUT-----
C      IF (NO .NE. 0 .OR. ICL .GE. 2) GO TO 10
C      WRITE (5,90)
C      IRET=1
C      RETURN
10    CCNTINUE
C      --TRANSFER FUNCTION CHECKS-----
C      IF (IE .EQ. 0) IE=6
C      EPS=10.**(-IE)
C      --CPEN LOOP TF-----
C      IF (ITF1 .EQ. 0 .OR. NC .NE. 0) GO TO 20
C      WRITE (5,100)
C      IRET=1
C      RETURN
20    IF (ITF3 .EQ. 0) GO TO 30
C      IF (IREG .EQ. 0 .AND. (NC .NE. 0 .AND. NG .NE. 0)) GO TO 30
C      WRITE (5,110)
C      IRET=1
C      RETURN
30    CCNTINUE
C      --NOISE TF-----
C      IF (ITF2 .EQ. 0) GO TO 40
C      IF (NG .NE. 0 .AND. NC .NE. 0) GO TO 40
C      WRITE (5,120)
C      IRET=1
C      RETURN
C      --DESTABILIZATION RESTRICTIONS-----
40    IF (IDSTAE .EQ. 0) GO TO 50
C      IF (NC .EQ. 0) GO TO 50
C      IF (NG .NE. 0) IREG=1
C      WRITE (5,130)
C      IF (IREG .EQ. 1) GO TO 50
C      IRET=1
C      RETURN
50    CONTINUE
C      --PSD INPUT-----
C      IF (IPSD .EQ. 0) GO TO 80
C      IF (IPSD .LT. 0 .OR. PSD .GT. 3) GO TO 60
C      IF (IYU .LT. 0 .OR. IYU .GT. 3) GO TO 60
C      IF (INCRM .LT. 0 .OR. INORM .GT. NG+NO) GO TO 60
C      GO TO 70
C      WRITE (5,140)
C      IRET=1
C      RETURN
60    IF (IREG .EQ. 0 .AND. NC .NE. 0) GO TO 80
C      WRITE (5,150)
C      IRET=1
C      RETURN
80    CONTINUE
C      RETURN
90    FORMAT (//,5X,49H H - MATRIX MUST BE INPUT, I.E. "NO" MUST BE > 0.
100   1,/)
110   1,10X,26H TO COMPUTE OPEN LOOP T. F. //)
110   1,FORMAT (//,5X,48H REGULATOR AND FILTER SYNTHESIS MUST BE REQUESTED,
110   1//,5X,44H IN THE SAME RUN TO COMPUTE COMPENSATOR T. F.,//,5X,47H I.E.
120   21,REG MUST = 0. ; "NC" AND "NG" MUST BE > 0. //)
120   1,FORMAT (//,5X,51H NOISE T. F. CALCULATED ONLY WHEN REGULATOR DESIGN
130   1ED,/,5X,47H I.E. IREG MUST = 1. ; "NC" AND "NG" MUST BE > 0. //)
130   1,FORMAT (//,5X,47H DESTABILIZATION OPTION DESIGNED FOR A REGULATOR //,
130   1,5X,38H OR FILTER BUT NOT BOTH SIMULTANEOUSLY.//,5X,55H IF "NG" > 0
140   2. THE REGULATOR OPTION IS AUTOMATICALLY SET! //)
140   1,FORMAT (//,5X,49H ***** INCONSISTENT PSD INPUT FLAGS ****)
1,/)

```


150 FORMAT (//,5X,44H BOTH A REGULATOR AND FILTER MUST BE RESIDENT,/,10
1X,42H TO COMPUTE THE PSD OF A CONTROLLED SYSTEM!,/,10X,42H I.E. IREG
2 MUST BE 0. AND "NC" MUST BE > 0.,//)
END


```

C=====
C SUBROUTINE INNER (NS,NC,NO,NG,N2,ACL,B,BA,CI,CR,CO,CWI,CWR,D,PBGC,
1 PBGE,G,GAM,GM,GN,HO,D1,D2,PRO,BE,SC,C,SC,AR,DI,11,21,X,WNOEM,WNO
2 RMI,DESTAL,AA,EM,CM,JCP,RES,AY,BB,CC,CP,GN,GV,HY,HU,DSSTORE,ISAF,IS
3 AH,ISAG,IGAM,IEET,PRTT,NRCW,NCCL)
C=====
C IMPLICIT REAL*8 (A-H,C-Z)
C
C----- DIMENSION ACL (NS,NS), B (NC,NC), EA (NS,NS), CI (NS), CR (NS), CO (NS,NS), CW
1 (NS), CR (NS), PBGC (NS,NO), G (NS,NS), EM (NS,NS), PRO (NS,NS
2), RC (NO,NC), SC (NS,NS), AR (N2), TI (N2), W11 (NS,NS), W21 (NS,NS), X (N2,N2)
3, GN (NS,NS), HO (NC,NS), D1 (N2), D2 (N2), RM (N2,N2), Q (NG,NG), D (NO,NC), GAM
4 (NS,NG), WORM (NS,NS), WORMI (NS,NS), DESTAB (NS), AA (NS,NS), BM (NS,NC),
5 CM (NO,NS), JCF (N2), RES (N2), AY (NC,NC), BE (N2), CC (N2), C2 (NS), GW (N2,NG)
6, GV (N2,NC), HY (NO, N2), BU (NC, N2), DSTORE (NS,NS), PBCT (16, 16)
C----- COMMON /PROG/ IOL, IQ, IR, ISS, IM, ITF1, ITF2, ITF3, IPDFW, IE, IDSTAB, IDEB
1UG, ISET, IREG, IFSD, IYU, INORM
C----- REAL*4 FMT(20)
C----- OUTPUT OPTIONS
C--- IOL=1 IF THE OPEN LOOP EIGENSYSTEM IS DESIRED--OTHERWISE IOL=0
C--- IQ=1 IF THE RMS VALUES OF THE CONTROL AND STATE ARE TO BE FOUND
C--- IB=0 IF OPTIMAL FILTER AND REGULATOR EIGENSYSTEMS ARE TO BE FOUND
C--- IR=1 IF EXTERNAL C MATRIX IS SUPPLIED
C--- IR=2 IF EXTERNAL K IS SUPPLIED
C--- IR=3 IF EXTERNAL C AND K ARE SUPPLIED
C--- ISS=1 IF STEADY STATE VALUES ARE TO BE DETERMINED
C--- IM=1 IF MODAL STATES DESIRED
C----- NSQ=NS*NS
N=NS
M=N2
CALL CHECK (EPS, NC, NG, NO, IRET)
IF (IRET.EQ.1) RETURN
IF (ISAT.EQ.1) GO TO 20
CALL READF (NS, ISAT, EA)
IF (IDSTAB.EQ.0) GO TO 10
WRITE (5, 1800)
CALL RDREAL (ANSR)
DSTAB=ANSR
DO 10 I=1, NS
DESTAB (I)=DSTAB
10 CONTINUE
GO TO 30
20 CALL SETUP (BA, G, GAM, NS, NG, NC)
30 CONTINUE
WRITE (6, 1380)
DO 40 I=1, NS
40 WRITE (6, 1390) (EA (I, J), J=1, NS)
IF (IDSTAB.EQ.0) GO TO 50
WRITE (6, 1480)
50 WRITE (6, 1390) (DESTAB (I), I=1, NS)
CONTINUE
C----- EIGENSYSTEM OF THE OPEN LOOP DYNAMICS-----
IF (IOL.EQ.0.AND.IQ.EQ.0) GO TO 90
IF (IOL.EQ.0.AND.NC.NE.0) GO TO 90
DO 60 I=1, NS
DO 60 J=1, NS
GN (I, J)=EA (I, J)
CALL BALANC (NS, NS, GN, LOW, IHIGH, D1)
CALL ORTEES (NS, NS, LCW, IHIGH, GN, D2)
CALL ORTPAN (NS, NS, LCW, IHIGH, GN, D2, SC)
CALL HCB2 (NS, NS, LOW, IHIGH, GN, CWR, CWI, SC, IERR)
IF (IERR.NE.0) CALL EREXIT (NS, GN, IERR)
CALL BALBAK (NS, NS, LOW, IHIGH, I1, NS, SC)
C----- NORMALIZE AND PRINT OPEN LOOP EIGENSYSTEM-----
IWRITE=1
CALL CNOEM (CWR, CWI, SC, NS, IWRITE, NSQ, DDD, D1, D2, WORM, WORMI, HC, CM,
1 NO, NS)
IF (IOL.EQ.2) RETURN
IF (IQ.EQ.0.OR.(NC.NE.0.OR.IDSTAB.GT.0)) GO TO 90
DO 70 I=1, NS
IF (CWR (I).LT.0.) GO TO 70
WRITE (5, 1490)
RETURN

```



```

70      CONTINUE
    IF (IOL.EQ.3) GO TO 130
    DO 80 I=1,NS
    DO 80 J=1,NS
80      W11(I,J)=SC(I,J)
    CALL LINV (NSQ,W11,NS,DDD,D1,E2)
90      CONTINUE
    IF (IDSTAB.EQ.0) GO TO 130
C----- FCBM U * DIAG (DESTAB) * U-INV-----
    DO 100 J=1,NS
100     DO 100 I=1,NS
    AA(I,J)=WNORM(I,J)*DESTAB(J)
    DO 120 I=1,NS
    DO 120 J=1,NS
    DDD=0. DO
    DO 110 K=1,NS
110     DDD=DDD+AA(I,K)*WNORM(I,K)
    DSTORE(I,J)=DDD
    BA(I,J)=BA(I,J)+DDD
120     CONTINUE
    CALL READH (NC,NS,ISAH,HO)
    WRITE (6,1440)
    DC 140 I=1,NO
140     WRITE (6,1390) (HC(I,J),J=1,NS)
    IF (IM.NE.1) GC TC 150
    CALL JCDE (WNCRM,HO,CM,NS,NC,NS,2)
150     CONTINUE
    IF (IDFW.EQ.0) GC TC 170
    CALL READD (NC,NC,D)
    WRITE (6,1470)
    DC 160 I=1,NO
160     WRITE (6,1390) (D(I,J),J=1,NC)
170     CONTINUE
    NOB=0
    IF (NC.EC.0) GC TC 590
    IF (IOL.EQ.3) GO TO 270
    IF (IR.NE.1.AND.IE.NE.3) GO TO 210
    IF (ISET.EQ.1) GO TO 180
    CALL READG (NS,NC,ISAG,G)
180     CONTINUE
    CALL READFB (NC,NS,FEGC)
    WRITE (6,1400)
    DO 190 I=1,NS
190     WRITE (6,1390) (G(I,J),J=1,NC)
    IF (IM.NE.1) GC TC 200
    CALL JCDE (WNCRMI,G,EM,NS,NS,NC,0)
200     CONTINUE
    GO TO 330
210     DO 220 I=1,NS
    DO 220 J=1,NS
220     RM(I+MH,J)=0.0
    CALL READAY (NO,AY)
    DO 240 I=1,NO
    DO 240 J=1,NS
    DDE=0. DO
    DO 230 K=1,NO
230     DDD=DDD+AY(I,K)*EC(K,J)
    AA(I,J)=DDD
    WRITE (6,1460)
    DO 250 I=1,NO
250     WRITE (6,1390) (AY(I,J),J=1,NC)
    DO 260 I=1,NS
    DO 260 J=1,NS
    DO 260 K=1,NO
260     RM(I+MH,J)=RM(I+MH,J)+AA(K,I)*EO(K,J)
    IF (ISET.EQ.1) GO TO 280
    CALL READG (NS,NC,ISAG,G)
280     CONTINUE
    IF (IOL.EC.3) GO TO 290
    CALL READE (NC,E)
290     WRITE (6,1400)
    DO 300 I=1,NS
300     WRITE (6,1390) (G(I,J),J=1,NC)
    IF (IM.NE.1) GC TC 310
    CALL JCDE (WNORMI,G,EM,NS,NS,NC,0)
310     CONTINUE

```



```

460  BA(I,I)=EA(I,I)-DESTAB(I)
470  IR=1
470  CCNTINUE
C-----CALCULATION OF FEEDBACK GAIN
C-----FEEDBACK GAINS--> U = -(BINVERSE)*GT*GN
C-----CALCULATE GT-
480  DO 490 I=1,NC
480  DO 490 J=1,NS
480  FRC(I,J)=0.00
480  DO 490 K=1,NC
480  PBC(I,J)=PBO(I,J)+G(K,I)*GN(K,J)
490  PBGC(I,J)=-PRO(I,J)/B(I,I)
490  IF (IDST.EQ.1) GO TO 500
C-----NORMALIZE AND PRINT OPT. REG. CLOSED LOOP EIGENSYSTEM
490  IWRITE=2
490  CALL CNORM (CWB,CWI,SC,NS,IWRITE,NSQ,DDD,D1,D2,WNORM,WNORMI,FBGC,
490  1AA,NC,NS)
C-----THE OPTIMUM FEEDBACK CONTROL GAINS-
500  WRITE (6,1580)
500  DC 510 I=1,NC
510  WRITE (6,1590) (FBGC(I,J),J=1,NS)
C-----COMPUTE MODAL C MATRIX OPEN LOOP U-INV SAVED IN WNORMI -----
510  IF (IM .NE. 1) GO TO 530
C-----IN COMPUTING MODAL C BECCMPUTE U OPEN LOOP SINCE WNORM USED TO STORE
C-----U & U-INV FOR CLOSED LOOP SYSTEMS: WNORMI USED TO SAVE U-INV OPEN LCCP
520  DO 520 I=1,NC
520  DO 520 J=1,NS
520  WNCRM(I,J)=WNCRM(I,J)
520  CALL MINV (NSC,WNCRM,NS,DDD,D1,D2)
520  CALL MODE (WNCRM,FBGC,AA,NS,NC,NS,3)
530  CONTINUE
C-----THE CLOSED LOOP DYNAMICS MATRIX-
540  DO 550 I=1,NS
540  DO 550 J=1,NS
540  SUM=0.00
540  DO 540 K=1,NC
540  SUM=SUM+G(I,K)*FBGC(K,J)
550  ACL(I,J)=EA(I,J)+SUM
550  WRITE (6,1600)
550  CALL 3ADRN (MH,MH,MH,5,ACL,4'(5(1X,1PD13.6)))
550  IF (IR.NE.1.AND.IB.NE.3) GO TO 590
550  DO 560 I=1,NS
550  DO 560 J=1,NS
560  GN(I,J)=ACL(I,J)
560  CALL BALANC (NS,NS,GN,LOW,IHIGH,D1)
560  CALL ORTHES (NS,NS,ICW,IHIGH,GN,D2)
560  CALL ORTFAN (NS,NS,LOW,IHIGH,GN,D2,SC)
560  CALL HQR2 (NS,NS,ICW,IHIGH,GN,CAR(CWI,SC,IERR)
560  IF (IERR .NE. 0) CALL EREXIT (NS,GN,IERR)
560  CALL BALBAK (NS,NS,LOW,IHIGH,D1,NS,SC)
C-----NORMALIZE AND PRINT CLOSED LOOP SUBOPT. REG. EIGENSYSTEM-----
560  IWRITE=3
560  CALL CNORM (CWB,CWI,SC,NS,IWRITE,NSQ,DDD,D1,D2,WNORM,WNORMI,FBGC,
560  1AA,NC,NS)
560  DO 570 I=1,NS
560  IF (CWB(I).LT.0.0) GO TO 570
560  WRITE (6,1610)
560  RETURN
570  CONTINUE
570  IF (IQ.NE.1) GO TO 590
570  DO 580 I=1,NS
570  DO 580 J=1,NS
580  W11(I,J)=SC(I,J)
580  CALL MINV (NSC,W11,NS,DDD,D1,D2)
590  NOB=NO
590  IF (NG.EQ.0) RETURN
600  IF (ISET.EQ.1) GO TO 610
600  CALL READG2 (NS,NG,IGAM,GAM)
610  CONTINUE
610  IF (IOL.EQ.3) GO TO 620
620  CALL READQ (NG,Q)
620  WRITE (6,1420)
620  DO 630 I=1,NS
630  WRITE (6,1590) (GAM(I,J),J=1,NG)

```



```

1NO,NS)
770  DO 780 I=1,MM
      DO 780 J=1,NO
      FEC(I,J)=HO(J,I)/RC(J,J)
      DO 790 I=1,MM
      DO 790 J=1,NO
      FBGE(I,J)=0.0
      DO 790 K=1,MM
790  PBGE(I,J)=FBGE(I,J)+GN(I,K)*FEC(K,J)
      IF (IDSTAE.EQ.1) GO TO 810
      WRITE(6,1670)
      CALL RAPRNT(MB,1E,MM,5,GN,4,'(5(1X,1PC13.6))')
      WRITE(6,1680)
      DO 800 I=1,MM
800  X(I,I)=DSQRT(GN(I,I))
      WRITE(6,1690) (X(I,I),I=1,MM)
810  WRITE(6,1630)
      DO 820 I=1,MM
820  WRITE(6,1640) (FEGE(I,J),J=1,NO)
C-----COMPUTE MODAL K MATRIX OPEN LOOP U-INV SAVED IN WNORMI -----
      IF (IM.NE.1) GC TO 830
      CALL MCDE(WNORMI,FEGE,AA,MM,MB,NO,4)
830  CONTINUE
C-----RESET FLAG AND F MATRIX FOR ITERATIVE DESTABILIZATION CASE-----
      IF (IDSTAB.EQ.0) GC TO 850
      DC 840 I=1,NS
      DO 840 J=1,NS
840  BA(I,J)=EA(I,J)-DSTORE(I,J)
      IR=2
850  CONTINUE
      DO 870 I=1,NS
      DC 870 J=1,NS
      SUM=0.0
      DC 360 K=1,NO
860  SUM=SUM+FEGE(I,K)*HC(K,J)
      FEC(I,J)=EA(I,J)-SUM
      WRITE(6,1650)
      CALL RAPRNT(NS,NS,NS,5,PB0,4,'(5(1X,1PC13.6))')
      IF (IR.LT.2) GO TO 890
      CALL BALANC(NS,NS,PRO,LOW,IHIGH,D1)
      CALL ORTRES(NS,NS,LOW,IHIGH,PRO,D2)
      CALL ORTRAN(NS,NS,LCW,IHIGH,PB0,D2,GM)
      CALL HQR2(NS,NS,LOW,IHIGH,PRO,CR,CI,GM,IERR)
      IF (IERR.NE.0) CALL BERExit(NS,PRO,IERR)
      CALL BALEAK(NS,NS,LCW,IHIGH,D1,NS,GM)
      WRITE(6,1560)
C-----NORMALIZE AND PRINT SUBOPT. ESTIMATOR EIGENSYSTEM-----
      IWRITE=5
      CALL CNORM(CR,CI,GM,NS,IWRITE,NSQ,DDD,D1,D2,WNORM,WNORMI,HO,AA,
      NC,NS)
      DO 880 I=1,NS
      IF (CR(I).LT.0.0) GO TO 880
      WRITE(5,1660)
      RETURN
880  CONTINUE
      GO TO 900
890  IP (10.EC.0) GC TO 1260
900  DO 910 I=1,NO
      DO 910 J=1,MM
      FEC(I,J)=0.0
      DO 910 K=1,NO
910  FEC(I,J)=HO(I,J)+RC(I,K)*FBGE(J,K)
      DO 920 I=1,MM
      DO 920 J=1,MM
      CQ(I,J)=0.0
      DO 920 K=1,NO
920  CQ(I,J)=CQ(I,J)-FEGE(I,K)*PRO(K,J)
930  CONTINUE
C-----THE RMS STATE AND CONTROL RESPONSES-----
      IR=IR+1
      GO TO (1C50,1C90,94C,940), IR
      DO 950 I=1,NS
      DO 950 J=1,NG
      X(I,J)=0.0
      DO 950 K=1,NG
950  X(I,J)=X(I,J)+GAM(I,K)*Q(K,J)

```



```

DO 970 I=1,NS
DO 970 J=1,NS
SUM=0.0
960 DO 960 K=1,NG
SUM=SUM-X(I,K)*GM(J,K)
PRO(I,J)=SUM+CQ(I,J)
FRC(J,I)=FRO(I,J)
CQ(I,J)=SUM
CQ(J,I)=SUM
W21(I,J)=GM(I,J)
W21(J,I)=GM(J,I)
CALL MINV (NS, W21, NS, DDD, D1, D2)
CALL SCOV (NS, GM, W21, CR, CI, NS, GM, W21, CR, CI, PRO, GN)
WRITE (6,1670)
CALL RABRT ('MH,MH,MH,5,GN,4,' (5(1X,1PD13.0)))
WRITE (6,1680)
DO 980 I=1,MH
980 X(I,I)=DSQRT(GN(I,I))
WRITE (6,1690) X(I,I), I=1,MH
IF (I.EQ.0) GC TC 1260
DO 1000 I=1,NC
DO 1000 J=1,NS
SUM=0.0
990 DO 990 K=1,NS
SUM=SUM+FEGC(I,K)*GN(K,J)
X(I,J)=SUM
DC 1020 I=1,NS
DC 1020 J=1,NS
SUM=0.0
1000 IF (NC.EQ.0) GC TC 1020
DO 1010 K=1,NC
SUM=SUM+G(I,K)*X(K,J)
PRO(I,J)=CQ(I,J)+SUM
CALL SCOV (NS, SC, W11, CWR, CWI, NS, GM, W21, CR, CI, PRO, BA)
IF (NC.EQ.0) GC TC 1040
DC 1030 I=1,NC
DC 1030 J=1,NS
W21(I,J)=C.0
DC 1030 K=1,NS
W21(I,J)=W21(I,J)+FBGC(I,K)*BA(J,K)
DC 1060 I=1,NS
DO 1060 J=1,NS
SUM=0.0
IF (NC.EQ.0) GC TC 1060
DC 1050 K=1,NC
1050 SUM=SUM+G(I,K)*W21(K,J)
1060 FRC(I,J)=SUM
DO 1070 I=1,NS
DO 1070 J=1,NS
FRC(I,J)=FRO(I,J)+CQ(I,J)+PRO(J,I)
PRO(J,I)=FRO(I,J)
CALL SCOV (NS, SC, W11, CWR, CWI, NS, SC, W11, CWR, CWI, PRO, CQ)
DO 1080 I=1,NS
DO 1080 J=1,NS
GM(I,J)=CC(I,J)-EA(I,J)-BA(J,I)+GN(I,J)
GM(J,I)=GM(I,J)
GO TO 1100
1090 CALL SCOV (NS, SC, W11, CWR, CWI, NS, SC, W11, CWR, CWI, CQ, GM)
1100 IF (NC.EQ.0) GC TC 1150
DO 1120 I=1,NS
DO 1120 J=1,NC
PRO(I,J)=C.0
DO 1110 K=1,NS
1110 PRO(I,J)=PRO(I,J)+GM(I,K)*FBGC(J,K)
CONTINUE
DO 1140 I=1,NC
DO 1140 J=1,NS
SC(I,J)=C.0
DO 1130 K=1,NS
1130 SC(I,J)=SC(I,J)+FEGC(I,K)*PRO(K,J)
CONTINUE
1140 IF (I.EQ.0) GO TO 1170
DO 1160 I=1,NS
DO 1160 J=1,NS
CQ(I,J)=GM(I,J)
GO TO 1190

```



```

1170  WRITE (6,1700)
      CALL RAPANT (ME,ME,ME,S,GM,4,'(5(1X,1PD13.6))')
      IF (IR.GT.2) GO TO 1190
      DO 1180 I=1,ME
      DC 1180 J=1,ME
      CQ(I,J)=GN(I,J)+GM(I,J)
1190  CONTINUE
      WRITE (6,1710)
      CALL RAPANT (ME,ME,ME,S,CQ,4,'(5(1X,1PD13.6))')
      IF (NC.EQ.0) GO TO 1210
      WRITE (6,1720)
      DC 1200 I=1,NC
1200  WRITE (6,1730) (SC(I,J),J=1,NC)
1210  DO 1220 I=1,NS
1220  CQ(I,I)=DSQRT(CQ(I,I))
      IF (NC.EC.0) GO TO 1240
      DO 1230 I=1,NC
1230  SC(I,I)=DSQRT(SC(I,I))
      WRITE (6,1740)
      DO 1250 I=1,NS
      IF (I.LE.NC) WRITE (6,1750) CC(I,I),SC(I,I)
      IF (I.GT.NC) WRITE (6,1750) CC(I,I),SC(I,I)
1250  CONTINUE
1260  IF (ITP3.EQ.0) GO TO 1290
C-----FORM COMPENSATOR FROM MEAS TC INPUT AND COMPUTE TF-----
      DO 1280 I=1,NS
      DO 1280 J=1,NS
      SUM=0.0
      DO 1270 K=1,NC
      SUM=SUM+FEGE(I,K)*HO(K,J)
1280  CQ(I,J)=ACL(I,J)-SUM
      WRITE (6,1760)
      ITFX=3
      IZERO=0
      CALL PF (NS,NS,NSC,CQ,AA,NO,FEGE,SM,NC,FBGC,CM,IZERO,D,BB,CC,CP,
      1WR,NI,CWR,CWI,SC,JCF,RES,D1,D2,DDD,EPS,ITP3,ITFX)
1290  CONTINUE
C-----COMPUTE PSD FUNCTIONS OF THE CONTROLLED SYSTEM-----
      IF (IPSD.EQ.0) GO TO 1310
      IF (IYU.LT.3) GO TO 1300
      CALL PSDCAL (M,NS,RM,X,NC,GW,GV,FBGC,NO,HY,HU,HO,FBGE,NG,
      1GAM,ACL,BA,WR,NI,D1,D2,JCF,RES,Q,RC,BE,CC,1,IPSD,INORM)
      CALL PSDCAL (M,NS,RM,X,NC,GW,GV,FBGC,NO,HY,HU,HO,FBGE,NG,
      1GAM,ACL,BA,WR,NI,D1,D2,JCF,RES,Q,RC,BE,CC,2,IPSD,INORM)
      GO TO 1310
1300  CALL PSDCAL (M,NS,RM,X,NC,GW,GV,FBGC,NO,HY,HU,HO,FBGE,NG,
      1GAM,ACL,BA,WR,NI,D1,D2,JCF,RES,Q,RC,BE,CC,IYU,IPSD,INORM)
1310  IF (ISS.EQ.0) RETURN
      IP (NC.NE.0) GO TO 1330
      DO 1320 I=1,NS
      DO 1320 J=1,NS
1320  ACL(I,J)=BA(I,J)
1330  CONTINUE
      CALL AINV (NSQ,ACI,NS,DDD,D1,D2)
      CALL READW (NG,WR)
      WRITE (6,1770) (WR(I),I=1,NG)
      WRITE (6,1780)
      DO 1340 I=1,NS
      WI(I)=0.0
      DO 1340 J=1,NG
      WI(I)=WI(I)+GAM(I,J)*WR(J)
      DO 1360 I=1,NS
      CR(I)=0.0
      DO 1350 J=1,NS
      CR(I)=CR(I)-ACI(I,J)*WI(J)
      WRITE (6,1790) CR(I)
      DO 1370 I=1,NC
      CI(I)=0.0
      DO 1370 J=1,NS
      CI(I)=CI(I)+FEGC(I,J)*CR(J)
      WRITE (6,1790) (CI(I),I=1,NC)
      RETURN
C-----
C670  FC6FORMAT (2X,1P6E14.6,/,2X,6D14.6)
1380  FORMAT (//,5X,45HCF6EN LOOP DYNAMICS MATRIX.....F...,//)
1390  FC6FORMAT (10(2X,0P6E11.4))

```



```

1400 FORMAT (//,5X,45HTHE CONTROL DISTRIBUTION MATRIX.....G...//)
1410 FORMAT (//,5X,45HTHE CONTROL CCST MATRIX.....B...//)
1420 FORMAT (//,5X,45HPROCESS NOISE DISTRIBUTION MATRIX.....GAMMA...//)
1430 FORMAT (//,5X,45HPOWER SPECTRAL DENSITY - PROCESS NOISE...Q...//)
1440 FORMAT (//,5X,45HMEASUREMENT SCALING MATRIX.....H...//)
1450 FORMAT (//,5X,45HPOWER SPECTRAL DENSITY-MEASUREMENT NOISE...R...//)
1460 FORMAT (//,5X,45HOUTUT CCST MATRIX.....A...//)
1470 FORMAT (//,5X,45HMEASUREMENT FEEDTHROUGH MATRIX.....D...//)
1480 FFORMAT (25X,23H...DESTABILIZATION CASE //,10X,39HTHE FOLLOWING
1 WING VALUES WILL BE ADDED DOWN //,10X,49HTHE DIAGONAL OF THE "F" MATRIX TO DESTABILIZE IT. //,10X,41HOPTIMAL GAINS FOR THE DESTABILIZED SYSTEM //,10X,39HARE THEN USED AS FIXED SUBOPTIMAL GAINS //,10X,28
2 FOR THE SYSTEM CALCULATIONS //)
1490 FFORMAT (//,43H PROGRAM TERMINATING DUE TO UNSTABLE SYSTEM)
1500 FORMAT (//,2X,31HOPEN LOOP TRANSFER FUNCTIONS...)
1510 FFORMAT (//,5X,32H EULER-LAGRANGE SYSTEM MATRIX...//)
1520 FORMAT (//,1X,43HEIGENVALUES AND EIGENVECTORS OF THE 2N X 2N //,2X,
145HEULER-LAGRANGE SYSTEM AFTER HQR2.....//)
1530 FFORMAT (1X,1P2E13.6)
1540 FORMAT (1X)
1550 FORMAT (//,2X,41HEIGENSYSTEM OF OPTIMAL REGULATOR.....//)
1560 FORMAT (//,2X,41HEIGENSYSTEM OF OPTIMAL ESTIMATOR.....//)
1570 FORMAT (//,5X,39HEIGENVECTORS FROM RGAINE PRIOR TO CNORM //)
1580 FORMAT (//,1X,57HTHE OPTIMAL FEEDBACK GAIN CONTROL MATRIX...C=BINV
1 *GT*S...//)
1590 FORMAT (1C(2X,1PD11.4))
1600 FORMAT (//,2X,45HTHE CLOSED LCCP DYNAMICS MATRIX.....F-G*C...//)
1610 FORMAT (//,60H PRCGRAM TERMINATING DUE TO UNSTABLE CLOSED LOOP
1 SYSTEM)
1620 FORMAT (//,2X,61HNOISE TRANSFER FUNCTIONS ,32H THROUGH THE CLOSED LOOP SYSTEM //)
1630 FORMAT (//,5X,45H FILTER STEADY STATE GAINS.....K...//)
1640 FORMAT (1X,2X,1P6D14.6)
1650 FORMAT (//,1X,43H THE CLOSED LCCP FILTER DYNAMICS MATRIX IS...//)
1660 FORMAT (//,1X,43H PROGRAM TERMINATING DUE TO UNSTABLE FILTER)
1670 FORMAT (//,5X,45H THE COVARIANCE OF THE ESTIMATION ERROR.....P...//)
1680 FORMAT (//,5X,45H RMS VALUES OF THE ESTIMATION ERROR.....//)
1690 FORMAT (15(1X,1P6D13.6)))
1700 FORMAT (//,5X,45H THE COVARIANCE OF THE ESTIMATE.....XHAT...//)
1710 FORMAT (//,5X,45H THE STATE COVARIANCE MATRIX.....X=XHAT + P...//)
1720 FORMAT (//,5X,45H THE CONTROL COVARIANCE.....U=C*XHAT*CT...//)
1730 FFORMAT (1P6D14.6)
1740 FORMAT (//,2X,18H STATE RMS RESPONSE, 20X,20H CONTROL RMS RESPONSE,/)
1750 FFORMAT (1X,1P15.7 25X,D15.7)
1760 FORMAT (//,5X,50H COMPENSATOR TRANSFER FUNCTIONS FROM MEAS. TO INPU
1 T, //,5X,52H.....U/Z = -C*(S1-F+G*C+K*H) INV*K...//)
1770 FORMAT (//,2X,46H STEADY DISTURBANCE VECTOR.....H...//)
1 10(1X,1P12.4)
1780 FORMAT (//,5X,45H STEADY STATE VALUES OF STATE VAR. ARE.....//)
1790 FORMAT (//,5X,47H STEADY STATE CONTROL IS.....//)
1 10(1X,1P12.4)
1800 FORMAT (//,5X,49H ENTER THE MAGNITUDE OF THE DESTABILIZATION VECTOR
1 //,8X,47H TO BE ADDED DOWN THE DIAGONAL OF THE "F"-MATRIX //,8X,18H
20 DESTABILIZE IT...//)
END

```



```
C=====
      SUBROUTINE RAPSN1 (NMAX,M,N,L,A,IDLIM,FM1)
      REAL*8 A(NMAX,N)
      DIMENSION FM1(IDLIM)
      NU=L
      DO 20 NL=1,N,L
      IF (NU.GT.N) NU=N
      DO 10 I=1,L
10      WRITE (6,FM1) (A(I,J),J=NL,NU)
      WRITE (6,30)
20      NU=NU+L
      RETURN
30      FORMAT (1X)
      END
```



```

C=====
C      SUBROUTINE RGAIN (M, NS, NC, NCB, WR, WI, VF, GN, W11, TCB, W21, LT, C, CI, CT, M
1HS, MT)
C      IMPLICIT REAL*8 (A-H, O-Z)
C      DIMENSION WR(M), WI(M), VF(M, M), GN(NS, NS)
C      DIMENSION W11(NS, NS), TCB(M, M), W21(NS, NS), LT(NS), MT(NS)
C      DIMENSION C(NS), CI(NS), CT(NS, NS)
C      K=1
C      KP=1
C      KN=1
C      NRZEV=0
C      NCPEZEV=0
10      IF (K.GT.M) GO TO 210
C-----C CHECK FOR ZIGVAL AT OR NEAR J-OMEGA AXIS TO INCLUDE IN E-L EIGSYS
C-----C TURN FIRST ONE POSITIVE AND SECOND ONE NEGATIVE
C-----EIGVRR=DAES(WR(K))
C-----IF (EIGVRR.GE.1.D-10) GO TO 60
C-----IF (WI(K)).EQ.40,20,40
20      NRZEV=NRZEV+1
C-----IF (NRZEV.GT.1) GO TO 30
C-----WR(K)=EIGVRR
C-----GC TO 80
30      WR(K)=-EIGVRR
C-----WRITE (6,290)
C-----GC TO 150
40      NCPEZEV=NCPEZEV+1
C-----IF (NCPEZEV.GT.1) GO TO 50
C-----WR(K)=EIGVRR
C-----WR(K+1)=EIGVRR
C-----GC TO 110
50      WR(K)=-EIGVRR
C-----WR(K+1)=-EIGVRR
C-----WRITE (6,300)
C-----GC TO 180
60      IF (WR(K)).EQ.140,70,70
70      IF (WI(K)).EQ.110,80,110
C-----EIGENVECTOR FOR REAL EIGENVALUE, POSITIVE
80      IF (NOB.EQ.0) GO TO 100
C-----DO 90 J=1,M
90      TCB(J, KP)=VF(J, K)
100     KP=KP+1
C-----K=K+1
C-----GO TO 10
C-----EIGENVECTOR FOR COMPLEX EIGENVALUE, POSITIVE REAL PART
110     IF (NOB.EQ.0) GO TO 130
C-----DO 120 J=1,M
120     PR=VF(J, K)
C-----PI=-VF(J, K+1)
C-----TCB(J, KP)=PR+PI
C-----TCB(J, KP+1)=PR-PI
130     KP=KP+2
C-----K=K+2
C-----GO TO 10
140     IF (WI(K)).EQ.180,150,180
C-----EIGENVECTOR FOR REAL EIGENVALUE, NEGATIVE REAL PART
150     C(KN)=WR(K)
C-----CI(KN)=WI(K)
C-----IF (NOB.NE.0) GO TO 170
C-----KNS=KN+NS
C-----DO 160 J=1,M
160     TCB(J, KNS)=VF(J, K)
170     KN=KN+1
C-----K=K+1
C-----GO TO 10
C-----EIGENVECTOR FOR COMPLEX EIGENVALUE, NEGATIVE REAL PART
180     RR=WR(K)
C-----RI=WI(K)
C-----C(KN)=RR
C-----C(KN+1)=RR
C-----CI(KN)=RI
C-----CI(KN+1)=-RI
C-----IF (NOB.NE.0) GO TO 200
C-----KNS=KN+NS
C-----DO 190 J=1,M

```



```

FF=VP(J,K)
FI=-VP(J,K+1)
TCB(J,KNS)=FR+FI
190 TCB(J,KNS+1)=FB-FI
RN=KN+2
200 K=K+2
GO TO 10
210 CONTINUE
210 IF (NOB.NE.0) GO TO 240
C-----FORMATION OF W11-----
220 DO 220 I=1,NS
220 DO 220 J=1,NS
220 W11(I,J)=TCB(I,J+NS)
220 CT(I,J)=W11(I,J)
C-----FORMATION OF W21-----
230 DO 230 I=1,NS
230 DO 230 J=1,NS
230 W21(I,J)=TCB(I+NS,J+NS)
240 IF (NOB.EQ.0) GO TO 260
240 DO 250 I=1,NS
240 DO 250 J=1,NS
250 W21(I,J)=TCB(I,J)
250 W11(I,J)=TCB(I+NS,J)
260 CONTINUE
C-----INVERT W11-----
C-----CALCULATE THE GAIN MATRIX-----
270 NSC=NS*NS
270 CALL MINV (NSC,W11,NS,DETC,LT,MT)
270 DO 270 IL=1,NS
270 DO 270 JL=1,NS
270 GN(IL,JL)=0.0D0
270 DO 270 KL=1,NS
270 GN(IL,JL)=GN(IL,JL)+W21(IL,KL)*W11(KL,JL)
270 IF (NOB.EQ.0) RETURN
270 DO 280 I=1,NS
270 DO 280 J=1,NS
280 CT(I,J)=W11(J,I)
280 RETURN
C-----FORMAT (1X,51H EULER-LAGRANGE EQUATIONS HAVE A REAL EIGENVALUE AT,
290 1148 OR NEAR ZERO./)
300 FORMAT (1X,49H EULER-LAGRANGE EQUATIONS HAVE A COMPLEX PAIR OF ,40
300 1EIGENVALUES AT OR NEAR THE J-CMEGA AXIS.)
END

```



```

C=====
C----- SUBROUTINE MINV (NSC,A,N,C,L,M)
C----- IMPLICIT REAL*8 (A-E,O-Z)
C----- DIMENSION A(NSC),I(N),M(N)
C----- DOUBLE PRECISION A,C,BIGA,HOLD
C----- NM=N*N
C----- D=1.0D0
C----- MK=-N
C----- DO 180 K=1,N
C----- NK=NK+N
C----- L(K)=K
C----- M(K)=K
C----- KK=NK+K
C----- BIGA=A(KK)
C----- DC 20 J=K,N
C----- IZ=N*(J-1)
C----- DC 20 I=K,N
C----- IJ=IZ+I
C----- IF (DAES(BIGA)-DAES(A(IJ))) 10,20,20
10    BIGA=A(IJ)
C----- L(K)=I
C----- M(K)=J
20    CCNTINUE
C----- -----INTERCHANGE ROWS-----
C----- J=L(K)
C----- IF (J-K) 50,5C,3C
30    RI=K-N
C----- DO 40 I=1,N
C----- KI=KI+N
C----- HOLD=-A(KI)
C----- JI=KI-K+J
C----- A(KI)=A(JI)
40    A(JI)=HOLD
C----- -----INTERCHANGE COLUMNS-----
50    I=M(K)
C----- IF (I-K) 60,60,60
60    JE=N*(I-1)
C----- DO 70 J=1,N
C----- JK=NK+J
C----- JI=JP+J
C----- HCOLD=-A(JK)
C----- A(JK)=A(JI)
70    A(JI)=HOLD
C----- -----DIVIDE COLUMN BY MINUS PIVOT-----
C----- ----- (VALUE OF PIVOT ELEMENT IS CONTAINED IN BIGA) -----
80    IF (BIGA) 100,9C,100
90    D=0.0D0
C----- RETURN
100   DC 120 I=1,N
C----- IF (I-K) 110,12C,110
110   IK=NK+I
C----- A(IK)=A(IK)/(-BIGA)
120   CCNTINUE
C----- -----REDUCE MATRIX-----
C----- DC 150 I=1,N
C----- IK=NK+I
C----- HOLD=A(IK)
C----- IJ=I-N
C----- DC 150 J=1,N
C----- IJ=IJ+N
C----- IF (I-K) 130,15C,130
130   IF (J-K) 140,15C,140
C----- KJ=IJ-I+K
140   A(IJ)=HOLD*A(KJ)+A(IJ)
150   CCNTINUE
C----- -----DIVIDE ROW BY PIVOT-----
C----- KJ=N-N
C----- DO 170 J=1,N
C----- KJ=KJ+N
C----- IF (J-K) 160,17C,160
160   A(KJ)=A(KJ)/BIGA
170   CCNTINUE
C----- -----PRODUCT OF PIVOTS-----
C----- D=E*BIGA
C----- -----REPLACE PIVOT BY RECIPROCAL-----
A(KK)=(1.0D0)/BIGA

```



```

180  CONTINUE
C-----FINAL ROW AND COLUMN INTERCHANGE-----
190  K=N
190  K=(K-1) 260,260,200
200  I=L(K)
200  I=(I-K) 230,230,210
210  JQ=N*(K-1)
210  JR=N*(I-1)
210  DO 220 J=1,N
210  JK=JQ+J
210  HOLD=A(JK)
210  JI=JR+J
210  A(JK)=-A(JI)
220  A(JI)=HOLD
230  J=M(K)
230  IF (J-K) 190,190,240
240  KI=K-N
240  DO 250 I=1,N
240  KI=KI+N
240  HOLD=A(KI)
240  JI=KI-K+J
240  A(KI)=-A(JI)
250  A(JI)=HOLD
250  GO TO 190
260  K='J
260  RETURN
260  END

```



```

C=====
 1  REAL*8  VL1(NL),VL2(NL),WL(NL,NL),WL1(NL,NL),X(NL,NR),Q(NL,NR),
 1  VR1(NR),VR2(NR),WR(NR,NR),WRI(NR,NR)
 10 DO 20 I=1,NL
 10 DO 20 J=1,NR
 10 X(I,J)=0.
 20 DO 20 II=1,NL
 20 X(I,J)=X(I,J)+WL1(I,II)*Q(II,J)
 20 DC 40 I=1,NL
 20 DO 40 J=1,NR
 20 Q(I,J)=0.
 30 DO 30 JJ=1,NR
 30 Q(I,J)=Q(I,J)+X(I,JJ)*WRI(J,JJ)
 40 CONTINUE
 40 I=1
 50 IF (VL2(I)) 60,110,60
 60 J=1
 70 IF (VR2(J)) 80,90,80
 80 A=VL1(I)+VR1(J)
 80 B=-2.*VL2(I)*VR2(J)
 80 C=A**2+VL2(I)**2+VR2(J)**2
 80 D=C**2-B**2
 80 K1=A*C/D
 80 K2=-(VR2(J)*C+VL2(I)*B)/D
 80 K3=-(VR2(J)*B+VL2(I)*C)/D
 80 K4=-A*B/D
 80 I=I+1
 80 J=J+1
 80 X(I,J)=+K1*Q(I,J)+K2*Q(I,J1)+K3*Q(I1,J)+K4*Q(I1,J1)
 80 X(I,J1)=-K2*Q(I,J)+K1*Q(I,J1)-K4*Q(I1,J)+K3*Q(I1,J1)
 80 X(I1,J)=-K3*Q(I,J)-K4*Q(I,J1)+K1*Q(I1,J)+K2*Q(I1,J1)
 80 X(I1,J1)=+K4*Q(I,J)-K3*Q(I,J1)-K2*Q(I1,J)+K1*Q(I1,J1)
 80 J=J+2
 90 GO TO 100
 90 A=VR1(J)+VL1(I)
 90 B=A**2+VL2(I)**2
 90 K1=A/B
 90 K2=VL2(I)/B
 90 X(I,J)=K1*Q(I,J)-K2*Q(I+1,J)
 90 X(I+1,J)=K2*Q(I,J)+K1*Q(I+1,J)
 90 J=J+1
100 IF (J.LE.NR) GO TC 70
100 I=I+2
110 GO TO 160
110 J=1
120 IF (VR2(J)) 130,140,130
130 A=VR1(J)+VL1(I)
130 B=A**2+VR2(J)**2
130 K1=A/B
130 K2=VR2(J)/B
130 X(I,J)=K1*Q(I,J)-K2*Q(I,J+1)
130 X(I,J+1)=K2*Q(I,J)+K1*Q(I,J+1)
130 J=J+2
140 GO TO 150
140 X(I,J)=Q(I,J)/(VR1(J)+VL1(I))
140 J=J+1
150 IF (J.LE.NR) GO TC 120
150 I=I+1
160 IF (I.LE.NL) GO TC 50
160 DO 170 I=1,NL
160 DO 170 J=1,NR
160 Q(I,J)=0.
160 DO 170 II=1,NL
160 Q(I,J)=Q(I,J)+WL(I,II)*X(II,J)
160 DO 190 I=1,NL
160 DO 190 J=1,NR
160 X(I,J)=0.
160 DC 180 J=1,NR
160 X(I,J)=X(I,J)+Q(I,JJ)*WR(J,JJ)
170 CONTINUE
170 RETURN
170 END

```



```

C=====
C      SUBROUTINE MODE (WNORM,G,GNORM,NS,N1,N2,ICON)
C
C      WNorm  TRANSFORMATION MATRIX U OR U-INV
C      NS    NO. OF STATE
C      NC    NO. OF INPUTS OR OUTPUTS
C      ICON  CONTROL FLAG TO INDICATE WHICH TRANSFORMATION
C              0 = MODAL G
C              1 = MODAL GAMMA
C              2 = MODAL H
C              3 = MODAL C
C              4 = MODAL K
C              5 = CONTROL EIGENVECTOR MATRIX
C              6 = MEASUREMENT EIGENVECTOR MATRIX
C=====
C      IMPLICIT REAL*8 (A-H,C-Z)
C      DIMENSION WNorm(NS,NS),G(N1,N2),GNORM(N1,N2)
C      DO 10 I=1,N1
C      DO 10 J=1,N2
C 10  WNorm(I,J)=0.
C      IFCINT=ICON+1
C      GO TO (20,20,90,90,20,90,90), IFCINT
C 20  DO 30 J=1,N2
C      DO 30 I=1,NS
C      DO 30 K=1,NS
C 30  GNORM(I,J)=GNORM(I,J)+WNORM(I,K)*G(K,J)
C      GC TO (40,70,90,90,80), IPOINT
C 40  WRITE (6,170)
C 50  DO 60 I=1,NS
C 60  WRITE (6,230) (GNORM(I,J),J=1,N2)
C      RETURN
C      WRITE (6,180)
C      GO TO 50
C 80  WRITE (6,240)
C      GO TO 50
C 90  GC 100 J=1,NS
C      DO 100 I=1,N1
C      DC 100 K=1,NS
C 100 GNORM(I,J)=GNORM(I,J)+G(I,K)*WNORM(K,J)
C      GC TO (110,110,110,120,110,130,140), IPOINT
C 110 WRITE (6,190)
C      GC TO 150
C 120 WRITE (6,200)
C 130 WRITE (6,210)
C      GO TO 150
C 140 WRITE (6,220)
C 150 DO 160 I=1,N1
C 160 WRITE (6,230) (GNORM(I,J),J=1,NS)
C      RETURN
C
C 170 FORMAT (//,5X,45HMODAL CONTROL DISTRIBUTION MATRIX....TI*G...//)
C 180 FORMAT (//,5X,5CHMODAL PROCESS NOISE DISTRIBUTION MATRIX...TI*GAM.
C 190 1.//)
C 190 FORMAT (//,5X,45HMODAL MEASUREMENT SCALING MATRIX...H(BAR)*T...//)
C 200 FORMAT (//,5X,45HTHE MODAL CONTROL GAINS.....C*T...//)
C 210 FORMAT (//,5X,45HCONTROL EIGENVECTOR MATRIX.....C*M...//)
C 220 FORMAT (//,5X,45HMEASUREMENT EIGENVECTOR MATRIX.....H(BAR)*M...//)
C 230 FORMAT (1X,(2X,1P6D14.6))
C 240 FORMAT (//,5X,45HMODAL FILTER STEADY STATE GAINS.....TI*K...//)
C      END

```



```

C=====
C      SUBROUTINE CNORM (WZ,WY,VEC,NS,IWRITE,NSQ,DDD,D1,D2,WNORM,WNORMI,H
C      1C,CM,N1,N2)
C
C      WZ(I)      REAL PART OF I-TH EIGENVALUE
C      WY(I)      COMPLEX PART OF I-TH EIGENVALUE
C      VEC        MATRIX OF RIGHT EIGENVECTORS STORED IN REAL FORM
C                  FFCM HQR2
C      NS         NO. OF STATES
C
C      IWRITE     FLAG TO CONTROL FORMATS FOR DIFFERENT EIGENSYSTEMS
C
C      WNORM     NORMALIZED MATRIX U OF RIGHT EIGENVECTORS STORED
C                  BY COOLUMNS IN REAL FORM
C      WNORMI    U-1 INVERSE 2*CONGUGATE OF LEFT EIGENVECTORS
C                  STORED BY ROW IN REAL FORM
C      NSQ,DDD,D1,D2 - ARGUMENTS PASSED TO KINV
C=====
C
C      IMPLICIT REAL*8 (A-H,O-Z)
C      REAL*8 FIELD,CMMA,SEMCOL,RIGHT,FMT
C      DIMENSION WZ(NS),WY(NS),VEC(NS,NS),WNORM(NS,NS),WNORMI(NS,NS),STOR
C      1E(6),D1(NS),D2(NS),FMT(14),HO(N1,N2),CM(N1,N2)
C      DATA FIELD/5H12.5/,CMMA/5H12.5/,SEMCOL/5H12.5/,RIGHT/1H//,FMT/
C      16H(1X,1E13*1H//SEMCOL/4H,':',/,RIGHT/1H)//,FMT/
C
C      ----- NORMALIZE COMPLEX EIGENVECTORS BY LARGEST ELEMENT -----
C
C      KK=0
C      LB=0
C      LC=0
C      DO 50 K=1,NS
C      IF (KK.EQ.1) GC TC 40
C      IF (DABS(WY(K)).LT.1.D-10) GO TO 50
C      LC=LC+1
C      EMAX=0.D0
C      DO 20 I=1,NS
C      CMOD=VEC(I,K)**2+VEC(I,K+1)**2
C      IF (CMOD-EMAX) 20,10,10
C      10 EMAX=CMOD
C      M=I
C      CONTINUE
C      VMR=VEC(M,K)
C      VMI=VEC(M,K+1)
C      DO 30 I=1,NS
C      VR=VEC(I,K)
C      VI=VEC(I,K+1)
C      VECRN=(VR*VMR+VI*VMI)/EMAX
C      VECIN=(-VR*VMI+VI*VMR)/EMAX
C      WNORM(I,K)=VECRN
C      WNORM(I,K+1)=VECIN
C      30 CONTINUE
C      KK=1
C      GO TO 50
C      40 KK=0
C      50 CONTINUE
C      ----- NORMALIZE REAL EIGENVECTORS BY THE TOTAL LENGTH -----
C      DO 80 K=1,NS
C      IF (DABS(WY(K)).GE.1.D-10) GO TO 80
C      LR=LR+1
C      REMOD=0.D0
C      DO 60 I=1,NS
C      REMOD=VEC(I,K)**2+REMOD
C      REMOD=DSQRT(REMOD)
C      DO 70 I=1,NS
C      RVEC=VEC(I,K)/REMOD
C      WNCRM(I,K)=RVEC
C      70 CONTINUE
C      80 CONTINUE
C      GO TO (90,100,110,120,130), IWRITE
C      90 WRITE(6,320)
C      GO TO 140
C      100 WRITE(6,330)
C      GO TO 140
C      110 WRITE(6,340)
C      GO TO 140
C      120 WRITE(6,350)

```



```

130      GC TO 140
140      WRITE (6,360)
         KK=0
         NPRTW=0
         NFMTW=1
         DO 180 I=1,NS
         IF (KK.EC.1) GC TO 170
         IF (DABS(WY(I)).GT.1.D-10) KK=1
C-----PRINT OUT NO MORE THAN 6 WORDS, NOT SEPARATING COMPLEX EIGVAL-----
         IF (NPRTW.LT.5.OR.(NFMTW.EQ.5.AND.KK.EQ.0)) GO TO 150
         FMT(NPRTW+1)=RIGHT
         WRITE(6,FMT) (STCRE(J),J=1,NFPTW)
         NPRTW=0
         NFMTW=1
150      NPRTW=NPRTW+1
         NFMTW=NFMTW+1
         IF (KK.EC.1) GC TO 160
         STORE(NPRTW)=WZ(I)
         FMT(NFMTW)=FIELD
         NFMTW=NFMTW+1
         FMT(NFMTW)=SEMCOL
         GC TO 180
160      STORE(NPRTW)=WZ(I)
         FMT(NFMTW)=FIELD
         FMT(NFMTW+1)=CCMMA
         STORE(NPRTW+1)=WY(I)
         FMT(NFMTW+2)=FIELD
         FMT(NFMTW+3)=SEMCCL
         NFMTW=NFMTW+3
         NFMTW=NPRTW+1
         GO TO 180
170      KK=0
180      CONTINUE
         FMT(NFMTW)=SEMCND
         FMT(NFMTW+1)=RIGHT
         WRITE(6,FMT) (STCRE(J),J=1,NFPTW)
         IF (I.EQ.1) GO TO 190
         WRITE(6,370)
         GC TO 200
190      WRITE(6,380)
200      CALL RAPRNT (NS,NS,NS,6,WNORM,4,'(6 (1X,1PD13.6)) ')
         GC TO (230,210,10,220,220),IWRITE
210      CALL MODE (WNORM,HO,CS,NS,N1,N2,5)
         GO TO 230
220      CALL MCDE (WNORM,HO,CM,NS,N1,N2,6)
230      GO TO (240,250,260,270,280),IWRITE
240      WRITE(6,390)
         GO TO 290
250      WRITE(6,400)
         GO TO 290
260      WRITE(6,410)
         GO TO 290
270      WRITE(6,420)
         GO TO 290
280      WRITE(6,430)
C-----SAVE U-INVERSE CFEN LOCP IN WNORMI-----
290      IF (IWRITE.GT.1) GC TO 310
         DO 300 I=1,NS
         DO 300 J=1,NS
300      WNORMI(I,J)=WNCFM(I,J)
         CALL MINV (NS,WNCRMI,NS,DDD,D1,D2)
         CALL RAPRNT (NS,NS,NS,6,WNORMI,4,'(6 (1X,1PD13.6)) ')
         RETURN
310      CALL MINV (NS,WNCRMI,NS,DDD,D1,D2)
         CALL RAPRNT (NS,NS,NS,6,WNORMI,4,'(6 (1X,1PD13.6)) ')
         RETURN
C-----FORMAT STATEMENTS-----
320      FORMAT (//1X,42H00EN LOOP EIGENVALUES. ....DET(SI-P)....//)
330      FFORMAT (//1X,46HC-LOCP OPTIMAL REG. E-VALUES...DET(SI-P+G*C)....//)
340      FORMAT (//1X,46HC-LOCP SUBOPT. REG. E-VALUES...DET(SI-P+G*C)....//)
350      FFORMAT (//1X,46HC-LOCP OPTIMAI EST. E-VALUES...DET(SI-P+K*H)....//)
360      FORMAT (//1X,46HC-LOCP SUBOPT. EST. E-VALUES...DET(SI-P+K*H)....//)
370      FORMAT (//1X,46HC0EN LOOP RIGHT EIGENVECTOR MATRIX.....T....//)
380      FFORMAT (//1X,46HC-LOOP RIGHT EIGENVECTOR MATRIX.....M....//)
390      FORMAT (//1X,46HC0EN LOOP LEFT EIGENVECTOR MATRIX.....P-INV....//)
400      FFORMAT (//1X,46HC-LOCP OPT. REG. LEFT E-VECTOR MATRIX..M-INV....//)

```



```
410  FORMAT (//1X,46HC-LOOP SUBOPT-REG. LEFT E-VECTOR MATRIX..M-INV,//)
420  FORMAT (//1X,46HC-1CCP OPT. FILTER LEFT E-VECTOR MATRIX..M-INV//)
430  FCRMAT (//1X,51HC-LOOP SUBOPT. FILTER LEFT E-VECTOR MATRIX..M-INV.
1. ,//)
      END
```



```

C=====
C SUBROUTINE TP (N,NM,NSQ,A,AA,M,B,BM,L,C,CM,IPDFW,D,BB,CC,CP,
1  EVR,EVI,PA,SI,SC,JCF,RES,D1,D2,DDD,EPS,ITF,ITFX)
1  IMPLICIT REAL*8 (A-H,O-Z)
1  DIMENSION A(N,N),AA(N,N),B(N,M),BM(N,M),C(L,N),CM(L,N),D(L,M),BB(N
1 ),CC(N),CE(N),EVR(N),EVI(N),PR(N),PI(N),SC(N,N),JCF(N),RES(N),D1(N
2 ),D2(N)
C--SAVE COMPUTATION ON CL AND CL SYS WITH MODAL WORK DONE IN OPTSYS-----
1  IF (ITFX .EQ. 1) GO TO 50
1  IF (ITFX .EQ. 2) GO TO 10
1  CALL POLES (N,NM,A,AA,M,B,L,C,PR,PI,D1,D2,JCF,SC)
C-----COMÉTIÉ MCAL MATRICES FOR RÉSIDUES-----
10  DO 20 I=1,N
10  DO 20 J=1,N
20  AA(I,J)=SC(I,J)
DO 30 I=1,L
DO 30 J=1,N
CM(I,J)=C(0,0)
DC 30 K=1,N
30  CM(I,J)=CM(I,J)+C(I,K)*AA(K,J)
CALL MINV (NSQ,AA,N,DDD,D1,D2)
DO 40 I=1,N
DO 40 J=1,M
BM(I,J)=0.00
DO 40 K=1,N
40  BM(I,J)=BM(I,J)+AA(I,K)*B(K,J)
CONTINUE
DO 60 I=1,M
DO 60 J=1,L
1  IF (ITF .NE. 3) CALL ZEROS (I,J,IPDFW,N,NM,A,AA,M,B,L,C,D,BB,CC,CP
1  ,EVR,EVI,D1,D2,EPS)
1  IF (ITF .NE. 2) CALL RESID (I,J,N,JCF,M,BM,L,CM,PR,PI,RES,BB,CC,1)
60  CONTINUE
1  RETURN
1  END

```



```

C=====
      SUBROUTINE POLES (N,SM,A,AA,E,E,L,C,EVB,EVI,D1,D2,JCF,SC)
      IMPLICIT REAL*8 (A-H,C-Z)
      DIMENSION A(N,N),AA(N,N),E(N,M),C(L,N),EVR(N),EVI(N),D1(N),D2(N),J
      JCF(N),SC(N,N)
      DC 10  I=1,N
      DO 10  J=1,N
      10  AA(I,J)=A(I,J)
      CALL BALANC (NM,N,AA,LOW,IGH,D1)
      CALL ORTHES (NM,N,LOW,IGH,AA,D2)
      CALL ORTRAN (NM,N,LOW,IGH,AA,D2,SC)
      CALL HQR2 (NM,N,LOW,IGH,AA,EVR,SC,IERR)
      IF (IERR .NE. 0) GO TO 30
      CALL BALBAK (NM,N,LOW,IGH,D1,N,SC)
      WRITE (6,40)
      20  WRITE (6,50) EVR(I),EVI(I)
      RETURN
      30  WRITE (5,60)
      RETURN
C-----
40  FORMAT (//,28H TF DENOMINATOR EIGENVALUES:,/)
50  FORMAT (32X,3H (F13.6 4H)+J(F13.6 4H))
60  FORMAT (35H FAILURE IN HQR2, CALCULATING POLES)
END

```



```

C=====
1  SUBROUTINE ZERCS (K1,K2,IFDFW,N,NM,A,AA,M,B,L,C,D,BB,CC,CP,EVR,EVI
1, D1,D2,EPSS)
1  IMPLICIT REAL*8 (A-H,C-Z)
1  DIMENSION A(N,N),AA(N,N),B(N,M),C(L,N),D(L,M),BB(N),CC(N),CP(N),EV
1  R(N),EVI(N),D1(N),D2(N)
1  DCUBLE PRECISICH SCL,DABS
1  DO 10 I=1,N
1  BB(I)=2*(I,K1)
1  CC(I)=C(K2,I)
10  DO 10 J=1,M
10  AA(I,J)=A(I,J)
10  WRITE(6,90) K1,K2
10  IF (IFDFW.EQ.0) GO TO 20
10  H=D(K2,K1)
10  IF (DABS(H).LE.EPSS) GO TO 20
10  JJ=N
10  GO TO 50
20  NN=N-1
20  DO 30 I=1,NN
20  H=SCL(N,BB,CC)
20  CALL CCMP(N,NM,AA,CC,CP)
20  IF (DABS(H).GT.EPSS) GO TO 40
30  CONTINUE
30  H=SCL(N,BB,CC)
30  WRITE(6,100) H
30  GO TO 70
40  JJ=N-I
40  WRITE(6,110) JJ,H
40  CALL ACOME(N,NM,AA,BB,CC,H)
40  CALL BALANC(NM,N,AA,LOW,IHIGH,D1)
40  CALL ORTHES(NM,N,LOW,IHIGH,AA,D2)
40  CALL HOR(NM,N,LOW,IHIGH,AA,EVS,EVI,IEEE)
40  IF (IERR.NE.0) GO TO 80
40  WRITE(6,120)
40  DC 50 I=1,N
60  WRITE(6,130) EVR(I),EVI(I)
70  RETURN
80  WRITE(5,140)
80  RETURN
C-----
90  FORMAT (//,17H TF FOR INPUT NO.,I3,15H AND OUTPUT NO.,I3,1H:)
100  FORMAT (//,5X,27HNO FINITE ZERCS. TF GAIN =,E12.4)
110  FORMAT (/,3X,20HORDER OF NUMERATOR = I3,9X,9HTF GAIN =,E12.4)
120  FORMAT (/,3X,57HNUMERATOR EIGENVALUES (INCLUDING EXTRANEOUS ZERO V
130  1ALUES):)
130  FORMAT (/-4X,1H(,F13.6,3H)+J(,F13.6,1H))
140  FORMAT (52H FAILURE IN HQR CALCULATING TRANSFER FUNCTION ZEROES)
140  END

```



```
C=====
      SUBROUTINE ACOMP (N, NM, A, B, C, E)
      REAL*8 A,B,C,E
      DIMENSION A(NM,N),B(N),C(N)
      DO 10 I=1,N
      DO 10 J=1,N
10      A(I,J)=A(I,J)-B(I)*C(J)/E
      RETURN
      END
```



```
C=====
      SUBROUTINE CCCMP (N, NM, A, C, CC)
      REAL*8 A, C, CC
      DIMENSION A(NM,N), C(N), CC(N)
      DO 10 I=1,N
      CC(I)=0.
10      DO 10 J=1,N
      CC(I)=CC(I)+C(J)*A(J,I)
10      DO 20 I=1,N
      C(I)=CC(I)
20      C(I)=C(I)
      RETURN
      END
```



```
C=====
FUNCTION SCL (N,B,C)
REAL*8 B,C,SCL
DIMENSION B(N),C(N)
SCL=0.
10 DO 10 I=1,N
    SCL=SCL+C(I) * B(I)
    RETURN
END
```



```

C=====
      SUBROUTINE RESID (K1,K2, N,JCF,M,BM,L,CM,PG,PI,RES,BB,CC,IPT)
      IMPLICIT REAL*8 (A-H,C-Z)
      DIMENSION JCF (N),EM(N,M),CM(L,N),PR (N),PI(N),RES(N),BB(N),CC(N),PR
      1T(4)
      DATA SN/8E*SIN(B*T/, R1/8H */, R2/8HEXP(A*T)/, ED/1H)/
      DATA ZERC/0.D0/, T1/4H*T**/, BLANK/8H /, CS/8H*COS(B*T/
C-----TEMPORARY MOD TILL JCF IS CALCULATED-----
      DO 10 I=1,N
10    JCF(I)=0
C-----TEMPORARY MCD-----
      I2 (IPT .EQ. 1) .NE.ITE (5,170)
      DO 20 I=1,N
      BE(I)=BM(I,K1)
20    CC(I)=CM(K2,I)
C-----LCOP THROUGH THE POLES-----
      I=0
30    I=I+1
      IF (I .GT. N) GO TO 160
      IF (JCF(I) .EQ. 1) GO TO 60
      IF (DABS(PI(I)) .LT. 1.D-10) GO TO 50
C-----COMPUTE SIMPLE COMPLEX POLE RESIDUES AND PRINT BOTH-----
      RES(I)=CC(I)*BE(I)+CC(I+1)*BB(I+1)
      RES(I+1)=CC(I)*BE(I+1)-CC(I+1)*BB(I)
      IF (IPT .EQ. 0) GC TC 40
      PRT(1)=BLANK
      PRT(2)=R2
      IF (PI(I) .EQ. 0.E0) PRT(2)=BLANK
      PRT(3)=CS
      PRT(4)=ED
      WRITE (6,180) PR(I),PI(I),RES(I),(PRT(J),J=1,4)
      I=I+1
      PRT(3)=SN
      WRITE (6,180) PR(I),PI(I),RES(I),(PRT(J),J=1,4)
      GO TO 30
40    I=I+1
      GO TO 30
50    CONTINUE
C-----COMPUTE SIMPLE REAL POLE RESIDUE-----
      RES(I)=CC(I)*BE(I)
      IF (IPT .EQ. 0) GC TO 30
      PRT(1)=R1
      PRT(2)=R2
      PRT(3)=BLANK
      PRT(4)=BLANK
      WRITE (6,180) PR(I),PI(I),RES(I),(PRT(J),J=1,4)
      GO TO 30
C-----LOOK AHEAD TO DETERMINE SIZE OF THE JORDAN BLOCK-----
60    K=1
      KT=N-I
      DO 70 J=I,KT
      IF (JCF(J) .EQ. 0) GO TO 80
70    K=K+1
80    CONTINUE
      IF (DABS(PI(I)) .LT. 1.D-10) GO TO 110
C-----COMPUTE REPEATED COMPLEX POLE AND PRINT OUT ALL FOUR-----
      K=1
      RES(I)=CC(I)*BE(I)+CC(I+1)*BB(I+1)+CC(I+2)*BB(I+2)+CC(I+3)*BB(I+3)
      RES(I+1)=CC(I)*BB(I+1)-CC(I+1)*BB(I)+CC(I+2)*BB(I+3)-CC(I+3)*BB(I+1)
12)
      RES(I+2)=CC(I)*BB(I+3)+CC(I+1)*BB(I+2)
      RES(I+3)=CC(I)*BB(I+3)-CC(I+1)*BB(I+2)
      IF (IPT .EQ. 0) GC TO 100
      PRT(1)=R1
      PRT(2)=R2
      IF (DABS(PR(I)) .GT. 1.D-10) GO TO 90
      PRT(1)=BLANK
      PRT(2)=BLANK
      PRT(3)=CS
      PRT(4)=ED
      WRITE (6,180) PR(I),PI(I),RES(I),(PRT(J),J=1,4)
      PRT(3)=SN
      I=I+1
      WRITE (6,180) PR(I),PI(I),RES(I),(PRT(J),J=1,4)
      PRT(1)=R1
      PRT(2)=R2

```



```

IF (DABS (PR(I)) .LT. 1.0-10) PRT(2) = BLANK
PRT(3) = CS
I=I+1
WRITE (6,190) PR(I), PI(I), RES(I), PRT(1), K, (PRT(J), J=2,4)
PRT(3) = SN
I=I+1
WRITE (6,190) PR(I), PI(I), RES(I), PRT(1), K, (PRT(J), J=2,4)
GC TO 30
100 I=I+3
GC TO 30
C----COMPUTE REPEATED REAL POLE RESIDUE AND PRINT OUT ALL K OF THEM-----
110 CONTINUE
KT=I+K-1
NN=0
DO 130 J=I,KT
NN=NN+1
RES(J)=ZERO
DO 120 JJ=J,KT
120 RES(J)=RES(J)+EB(JJ)*CC(JJ-NN+1)
CONTINUE
130 IF (IPT .EQ. 0) GC TO 150
NN=0
PRT(1)=T1
PRT(2)=R2
PRT(3)=BLANK
PRT(4)=BLANK
DO 140 J=I,KT
140 WRITE (6,190) PR(J), PI(J), RES(J), PRT(1), NN, (PRT(JJ), JJ=2,4)
NN=NN+
GC TO 30
150 I=KT
GO TO 30
160 CONTINUE
160 RETURN
C-----
170 FORMAT (//,3X,22HRESIDUES AT THE POLES:/,T16,9HP O L E S,T41,15HR
1E 3 I D U E S /,T9,7HREAL(A),126,7HIMAG(S))
180 FORMAT (/,4X,1H(,F13.6,4H)+J(,F13.6,1H),4X,1H(,F13.6,1H),3A8,A1)
190 FORMAT (/,4X,1H(,F13.6,4H)+J(,F13.6,1H),4X,1H(,F13.6,1H),A4,12,2X,
12A8,A1)
END

```



```

C----- SUBROUTINE BALANC (NM,N,A,LOW,IGH,SCALE)
      INTEGER I,J,K,L,M,N,CG,NM,IGH,LCW,IEXC
      REAL*8 A(NM,N),SCALE(N)
      REAL*8 C,F,G,R,S,E2,RADIX
      REAL*8 DABS
      LOGICAL NOCONV
      DATA RADIX/Z42100000C0000000/
C----- B2=BAD IX*RADIX
      K=1
      L=N
      GO TO 60
C----- IN-LINE PROCEDURE FOR ROW AND COLUMN EXCHANGE
10     SCALE(M)=J
      IF (J.EQ.M) GO TO 40
      DO 20 I=1,L
      F=A(I,J)
      A(I,J)=A(I,M)
      A(I,M)=F
20     CONTINUE
      DC 30 I=K,N
      F=A(J,I)
      A(J,I)=A(M,I)
      A(M,I)=F
30     CONTINUE
40     GO TO (5C 90), IEXC
C----- SEARCH FOR ROWS ISOLATING AN EIGENVALUE AND PUSH THEM DOWN
50     IF (L.EQ.1) GO TO 230
      L=L-1
60     DO 80 JJ=1,L
      J=L+1-JJ
      DO 70 I=1,L
      IF (I.EQ.J) GC TO 70
      IF (A(J,I).NE.0.0D0) GO TO 80
70     CONTINUE
      M=L
      IEXC=1
      GO TO 10
80     CONTINUE
      GO TO 100
C----- SEARCH FOR COLUMNS ISOLATING AN EIGENVALUE AND PUSH THEM LEFT
90     K=K+1
100    DG 120 J=K,L
      DO 110 I=K,L
      IF (I.EQ.J) GC TO 110
      IF (A(I,J).NE.0.0D0) GO TO 120
110    CONTINUE
      M=K
      IEXC=2
      GO TO 10
120    CONTINUE
C----- NOW BALANCE THE SUBMATRIX IN ROWS K TO L
130    DC 130 I=K,L
      SCALE(I)=1.0D0
C----- ITERATIVE LOOP FOR NORM REDUCTION
140    NOCONV=.FALSE.
      DO 220 I=K,L
      C=0.0D0
      E=0.0D0
      DO 150 J=K,L
      IF (J.EC.I) GO TO 150
      C=C+DABS(A(J,I))
      R=R+DABS(A(I,J))
150    CONTINUE
C----- GUARD AGAINST ZERO C OR R DUE TO UNDERFLOW
      IF (C.EC.0.0D0 .OR. R.EQ.0.0D0) GO TO 220
      G=R/RADIX
      F=1.0D0
      S=C+R
160    IF (C.GE.G) GO TO 170
      F=F*RADIX
      C=C*B2
      GC TO 160
170    G=R*RADIX
180    IF (C.LT.G) GO TO 190

```


$F = F / \text{RADIX}$
 $C = C / 92$
GO TO 180

C----- NOW BALANCE -----

```
190 IF ((C + F) / F .GE. 0.95) GO TO 220
G=1.0D0/F
SCALE(I)=SCALE(I)*F
NOCONV=.TRUE.
DC 200 J=K,N
200 A(I,J)=A(I,J)*G
DC 210 J=1,L
210 A(J,I)=A(J,I)*F
220 CONTINUE
IF (NOCONV) GO TO 140
230 LC4=K
IGH=L
RETURN
END
```



```

C=====
      SUBROUTINE ORTHES (NM,N,LOW,IGH,A,ORT)
      INTEGER I,J,M,N,I,JJ,LA,MP,NM,IGH,KE1,LOW
      REAL*8 A(NM,N),ORT(IGH)
      REAL*8 P,G,H,SCALE
      REAL*8 DSQRT,DAES,DSIGN
      LA=IGH-1
      KE1=LOW+1
      IF (LA .LT. KE1) GO TO 100
      DO 40 M=KE1,LA
      H=0.0D0
      ORT(M)=0.0D0
      SCALE=C.CDO
C-----SCALE COLUMN (ALGOL TOL THEN NOT NEEDED) -----
      DO 10 I=M,IGH
      SCALE=SCALE+DAES(A(I,M-1))
      IF (SCALE .EQ. C.CDO) GO TO 90
      M=M+IGH
      DO 20 II=M,IGH
      I=MP-II
      ORT(I)=A(I,M-1)/SCALE
      H=H+ORT(I)*ORT(I)
      20 CONTINUE
      G=-DSIGN(DSQRT(H),ORT(M))
      H=H-ORT(M)*G
      ORT(M)=ORT(M)-G
C-----FORM (I-(U*UT)/H) * A -----
      DO 50 J=M,N
      F=0.0D0
      DO 30 II=M,IGH
      I=MP-II
      P=F+ORT(I)*A(I,J)
      30 CONTINUE
      F=F/H
      DO 40 I=M,IGH
      A(I,J)=A(I,J)-P*CST(I)
      40 CONTINUE
C-----FORM (I-(U*UT)/H) *A*(I-(U*UT)/H) -----
      DC 80 I=1,IGH
      F=C.CDO
      DC 60 JJ=M,IGH
      J=MP-JJ
      F=F+ORT(J)*A(I,J)
      60 CONTINUE
      F=F/H
      DO 70 J=M,IGH
      A(I,J)=A(I,J)-F*CST(J)
      70 CONTINUE
      ORT(M)=SCALE*CST(M)
      A(M,M-1)=SCALE*G
      80 CONTINUE
      90 RETURN
      100 END

```



```

C=====
      SUBROUTINE ORTRAN (NM,N,LCW,IGH,A,ORT,Z)
      INTEGER I,J,N,KL,M,MP,NM,IGH,LCW,MP1
      REAL*8 A(NM,IGH),ORT(IGH),Z(NM,N)
      REAL*8 G
C-----INITIALIZE Z TO IDENTITY MATRIX-----
      DO 20 I=1,N
      DO 10 J=1,N
10      Z(I,J)=0.0D0
      Z(I,I)=1.0D0
20      CONTINUE
      KL=IGH-LCW+1
      IF (KL.LT.1) GO TO 80
      DO 70 JM=1,KL
      MP=IGH-MM
      IF (A(MP,MP-1).EQ. 0.0D0) GO TO 70
      MP1=MP+1
      DO 30 I=MP1,IGH
30      ORT(I)=A(I,MP-1)
      DO 60 J=MP,IGH
      G=0.0D0
      DO 40 I=MP,IGH
40      G=G+ORT(I)*Z(I,J)
C-----DIVISOR BELOW IS NEGATIVE OF H FORMED IN ORTHES.-----
C-----DOUBLE DIVISION AVOIDS POSSIBLE UNDERFLOW-----
      G=(G / ORT(MP))/A(MP,MP-1)
      DO 50 I=MP,IGH
50      Z(I,J)=Z(I,J) + G*ORT(I)
      CONTINUE
70      RETURN
80      END

```



```

SUBROUTINE HQZ2 (NM,N,LOW,IGH,H,WR,WI,Z,IERR)
  INTEGER I,J,K,L,N,EN,II,JJ,IL,MM,NA,NN,IGH,ITS,LOW,MP2,ENM2,I
  10 IERR
  REAL*8 H (NM,N),WR(N),WI(N),Z(NM,N)
  REAL*8 P,Q,R,S,T,X,Y,RA,SA,VI,VR,ZZ,NCRM,MACHEP
  REAL*8 DSQRT,LABS,CSIGN
  INTEGER MINO
  LCGICAL NCLLAS
  COMPLEX *16Z3
  COMPLEX *16DCMPLX
  REAL*8 DREAL,DIMAG

C-----STATEMENT FUNCTIONS ENABLE EXTRACTION OF REAL AND IMAGINARY
C-----PARTS OF DOUBLE PRECISION COMPLEX NUMBERS-----
  DREAL(Z3)=Z3
  DIMAG(Z3)=(0.0D0-1.0D0)*Z3
  DATA MACHEP/23416CC0000000000/
  IERR=0
  NCRM=0.0D0
  K=1

C-----STORE FCOTS ISOLATED BY BALANC AND COMPUTE MATRIX NORM-----
  DO 20 I=1,N
  DO 10 J=K,N
  10 NORM=NORM+DABS (H (I,J))
  K=I
  IF (I .GE. LOW .AND. I .LE. IGH) GO TO 20
  WR(I)=H(I,I)
  WI(I)=0.0D0
  20 CONTINUE
  EN=IGH
  T=0.0D0

C-----SEARCH FOR NEXT EIGENVALUES-----
  30 IF (EN .LT. LOW) GO TO 290
  ITS=0
  NA=EN-1
  ENM2=NA-1
  40 DO 50 LL=LOW,EN
  L=EN+LOW-LL
  IF (L .EQ. LOW) GC TC 60
  S=DABS (H (L-1,L-1))+CABS (H (L,L))
  IF (S .EQ. 0.0D0) S=NORM
  IF (DABS (H (L,L-1)) .LE. MACHEP * S) GO TO 60
  50 CONTINUE

C-----LOOK FOR SINGLE SMALL SUB-DIAGONAL ELEMENT-----
  60 X=H(EN,EN)
  IF (L .EQ. EN) GO TO 220
  Y=H(NA,NA)
  W=H(EN,NA)*H(NA,EN)
  IF (L .EQ. NA) GO TO 230
  IF (ITS .EQ. 30) GO TO 500
  IF (ITS .NE. 10 .AND. ITS .NE. 20) GC TC 80

C-----FORM SHIFT-----
  T=T+X
  DO 70 I=LOW,EN
  H(I,I)=H(I,I)-X
  S=DABS (H(EN,NA))+CAES (H(NA,ENM2))
  X=0.75D0*S
  Y=X
  W=-0.4375D0*S*S
  80 ITS=ITS+1

C-----LOOK FOR TWO CONSECUTIVE SMALL SUB-DIAGONAL ELEMENTS.-----
  DO 90 MM=L,ENM2
  N=ENM2+L-MM
  ZZ=H(M,M)
  R=X-ZZ
  S=Y-ZZ
  P=(R * S - Z)/H(M+1,M)+H(M,M+1)
  Q=H(M+1,M+1)-ZZ-R-S
  R=H(M+2,M+1)
  S=DABS (P)+DABS (Q)+CABS (R)
  P=P/S
  Q=Q/S
  R=R/S
  IF (M .EQ. L) GO TO 100
  IF (DABS (H(M,M-1)) * (DABS (Q) + DABS (R)) .LE. MACHEP * DABS (P))

```



```

1 * (DABS(H(M-1,M-1)) + DABS(ZZ) + DAES(H(M+1,M+1))) GO TO 100
90  CONTINUE
100  MP2=1+2
    DC 110 I=MP2, EN
    H{I,I-2}=0.0D0
    IF (I .EQ. MP2) GO TO 110
    H{I,I-3}=0.0D0
110  CONTINUE
C-----DOUBLE QR STEP INVOLVING ROWS L TO EN AND COLUMNS M TO EN-----
    DO 210 K=M,NA
    NOTLAS=N-E,NA
    IF (K .EQ. M) GO TO 120
    P=H(K,K-1)
    Q=H(K+1,K-1)
    R=0.0D0
    IF (NOTLAS) R=H(K+2,K-1)
    X=DABS(P)+DABS(Q)+DAES(R)
    IF (X .EQ. 0.0D0) GO TO 210
    P=P/X
    Q=Q/X
    R=R/X
120  S=DSIGN(DSQRT(P*P+Q*C+R*R),P)
    IF (K .EQ. M) GO TO 130
    H(K,K-1)=-S*X
    GO TO 140
130  IF (L .NE. M) H(K,K-1)=-H(K,K-1)
140  P=P+S
    X=P/S
    Y=Q/S
    ZZ=R/S
    Q=Q/P
    R=R/P
C-----ROW MODIFICATION-----
    DC 160 J=K,N
    P=H(K,J)+C*H(K+1,J)
    IF (.NOT. NOTLAS) GO TO 150
    P=P+R*H(K+2,J)
    H(K+2,J)=H(K+2,J)-P*ZZ
    H(K+1,J)=H(K+1,J)-P*Y
    H(K,J)=H(K,J)-P*X
150  CONTINUE
    J=MIN0(EN,K+3)
C-----COLUMN MODIFICATION-----
    DO 180 I=1,J
    P=Y*H(I,K)+Y*H(I,K+1)
    IF (.NOT. NOTLAS) GO TO 170
    P=P+ZZ*H(I,K+2)
    H(I,K+2)=H(I,K+2)-P*R
    H(I,K+1)=H(I,K+1)-P*Q
    H(I,K)=H(I,K)-P*Y
170  CONTINUE
    J=MIN0(EN,K+3)
C-----ACCUMULATE TRANSFORMATIONS-----
    DO 200 I=LOW,IGH
    P=X*Z(I,K)+Y*Z(I,K+1)
    IF (.NOT. NOTLAS) GO TO 190
    P=P+ZZ*Z(I,K+2)
    Z(I,K+2)=Z(I,K+2)-P*B
    Z(I,K+1)=Z(I,K+1)-P*Q
    Z(I,K)=Z(I,K)-P*Y
200  CONTINUE
    CONTINUE
    GO TO 40
C-----ONE ROOT FOUND-----
220  H(EN,EN)=X+T
    WH(EN)=H(EN,EN)
    WI(EN)=0.0D0
    EN=NA
    GO TO 30
C-----TWO ROOTS FOUND-----
230  P=(Y-X)/2.0D0
    Q=P*P+W
    ZZ=DSQRT(DABS(Q))
    H(EN,EN)=X+T
    X=H(EN,EN)
    H(NA,NA)=Y+T
    IF (Q .LT. 0.0D0) GO TO 270

```



```

C-----REAL FAIR-----
ZZ=R+DSIGN(ZZ,E)
WR(NA)=X+ZZ
WE(EN)=WR(NA)
IP(ZZ,NE,0.0D0) WR(EN)=X-W/ZZ
WI(NA)=0.0D0
WI(EN)=C.CD0
X=H(EN,NA)
S=DABS(X)+DABS(ZZ)
P=X/S
Q=ZZ/S
R=DSQRT(E*P+Q*C)
P=P/R
Q=C/R
C-----ROW MODIFICATION-----
DO 240 J=NA,N
ZZ=H(NA,J)
H(NA,J)=C*ZZ+P*H(EN,J)
H(EN,J)=C*H(EN,J)-P*ZZ
240 CONTINUE
C-----COLUMN MODIFICATION-----
DO 250 I=1,EN
ZZ=H(I,NA)
H(I,NA)=C*ZZ+P*H(I,EN)
H(I,EN)=C*H(I,EN)-P*ZZ
250 CONTINUE
C-----ACCUMULATE TRANSFORMATIONS-----
DO 260 I=LOW,IGH
ZZ=Z(I,NA)
Z(I,NA)=C*ZZ+E*Z(I,EN)
Z(I,EN)=C*Z(I,EN)-E*ZZ
260 CONTINUE
GC TO 280
C-----COMPLEX FAIR-----
270 WE(NA)=X+P
WR(EN)=X+P
WI(NA)=ZZ
WI(EN)=-ZZ
280 EN=EN-2
GO TO 30
C-----ALL FACTS FOUND. BACKSUBSTITUTE TO FIND-----
C-----VECTORS OF UPPER TRIANGULAR FORM-----
290 IF (NORM.EQ.0.0D0) GO TO 510
DO 450 NN=1,N
EN=N+1-NN
P=WR(EN)
Q=WI(EN)
NA=EN-1
IF (Q) 370,300,450
C-----REAL VECTOR-----
300 M=EN
H(EN,EN)=1.0D0
IF (NA.EQ.0) GO TO 450
DO 360 II=1,NA
I=EN-II
W=H(I,I)-P
R=H(I,EN)
IF (M.GT.NA) GO TO 320
DO 310 J=M,NA
R=R+H(I,J)*H(J,EN)
IF (WI(I).GE.0.0D0) GO TO 330
ZZ=W
S=R
GO TO 360
330 M=I
IF (WI(I).NE.0.CD0) GO TO 340
T=W
IF (W.EQ.0.0D0) T=MACHEP*NOEM
H(I,EN)=-R/T
GO TO 360
C-----SOLVE REAL EQUATIONS-----
340 X=H(I,I+1)
Y=H(I+1,I)
Q=(WR(I)-P)*(WR(I)-P)+WI(I)*WI(I)
T=(X-S-ZZ*R)/Q
H(I,EN)=T

```



```

IF (DABS (X) .LE. DABS (ZZ)) GO TO 350
H(I+1,EN)=(-R - W * I)/X
GO TO 360
350 H(I+1,EN)=(-S - Y * I)/ZZ
360 CONTINUE
C-----END REAL VECTOR-----
C-----GO TO 450
C-----COMPLEX VECTOR-----
370 M=NA
C-----LAST VECTOR COMPONENT CHOSEN IMAGINARY SO THAT-----
C-----EIGENVECTOR MATRIX IS TRIANGULAR-----
IF (DABS (H(EN,NA)) .LE. DABS (H(NA,EN))) GO TO 380
H(NA,NA)=C/H(EN,NA)
H(NA,EN)=-(H(EN,EN) - P)/H(EN,NA)
GO TO 390
380 Z3=DCMPLX (0.0D0, -H(NA,EN))/DCMPLX (H(NA,NA) - P, 0)
H(NA,NA)=CREAL (Z3)
H(NA,EN)=DIMAG (Z3)
390 H(EN,NA)=C.0D0
H(EN,EN)=1.0D0
ENM2=NA-1
IF (ENM2 .EQ. 0) GO TO 450
DO 440 II=1,ENM2
I=NA-II
W=H(I,I)-P
RA=0.5D0
SA=H(I,EN)
DO 400 J=M,NA
RA=RA+H(I,J)*H(J,NA)
SA=SA+H(I,J)*H(J,EN)
400 CONTINUE
IF (WI(I) .GE. 0.0D0) GO TO 410
ZZ=1
B=BA
S=SA
GO TO 440
410 M=I
IF (WI(I) .NE. 0.0D0) GO TO 420
Z3=DCMPLX (-RA, -SA)/DCMPLX (1, Q)
H(I,NA)=CREAL (Z3)
H(I,EN)=DIMAG (Z3)
GO TO 440
C-----SOLVE COMPLEX EQUATIONS-----
420 X=H(I,I+1)
Y=H(I+1,I)
VR=(WR(I) - P)*(WR(I) - P)+WI(I)*WI(I)-Q*Q
VI=(WR(I) - P)*2.0D0*Q
IF (VR .EQ. 0.0D0 .AND. VI .EQ. 0.0D0) VR=MACHEP*NORM*(DABS (W) + D
1ABS (Q) + DABS (X) + DABS (Y) + DABS (ZZ))
Z3=DCMPLX (X*VR-ZZ*RA+Q*SA, X*S-ZZ*SA-Q*RA)/DCMPLX (VR, VI)
H(I,NA)=CREAL (Z3)
H(I,EN)=DIMAG (Z3)
IF (DABS (X) .LE. DABS (ZZ) + DABS (Q)) GO TO 430
H(I+1,NA)=(-RA - W * H(I,NA) + Q * H(I,EN))/X
H(I+1,EN)=(-SA - W * H(I,EN) - Q * H(I,NA))/X
GO TO 440
430 Z3=DCMPLX (-R - Y*H(I,NA), -S - Y*H(I,EN))/DCMPLX (ZZ, Q)
H(I+1,NA)=DREAL (Z3)
H(I+1,EN)=DIMAG (Z3)
440 CONTINUE
C-----END COMPLEX VECTOR-----
450 CONTINUE
C-----END BACK SUBSTITUTION. VECTORS OF ISOLATED ROOTS-----
DO 470 I=1,N
IF (I .GE. LOW .AND. I .LE. IGH) GO TO 470
DO 460 J=I,N
460 Z(I,J)=H(I,J)
470 CONTINUE
C-----MULTIPLY BY TRANSFORMATION MATRIX TO GIVE-----
C-----VECTORS OF ORIGINAL FULL MATRIX.-----
DO 490 JJ=LOW,N
J=N+LOW-W-JJ
M=SIGN0 (J, IGH)
DO 490 I=LOW,IGH
ZZ=0.0D0
DO 480 K=LOW,M

```



```
480  ZZ=ZZ+Z(I,K)*H(K,J)
490  Z(I,J)=ZZ
500  CONTINUE
      GO TO 510
C-----SET ERROR --> NO CONVERGENCE TO AN EIGENVALUE AFTER 30 ITERATIONS-----
C-----EIGENVALUE AFTER 30 ITERATIONS-----
500  IERR=EN
510  RETURN
      END
```



```

C=====
C      SUBROUTINE ZBALAK (NM,N,LOW,IGH,SCALE,M,Z)
C      INTEGER I,J,K,Z,N,II,NM,IGH,LCW
C      REAL*8 SCALE(N),Z(NM,M),S
C      IF (M.EQ.0) GO TO 60
C      IF (IGH.EQ. LCW) GO TO 30
C      DO 20 I=LCW,IGH
C         S=SCALE(I)
C===== LEFT HAND EIGENVECTORS ARE BACK TRANSFORMED-----
C===== IF THE FOREGOING STATEMENT IS REPLACED BY -----
C===== S=1.0DC/SCALE (I). -----
C
10      DC 10 J=1,M
20      Z(I,J)=Z(I,J) *S
CONTINUE
30      DO 50 II=1,N
      I=II
      IF (I .GE. LOW .AND. I .LE. IGH) GO TO 50
      IF (I .LT. LOW) I=LOW-II
      K=SCALE(I)
      IF (K .EQ. I) GO TO 50
      DO 40 J=1,M
      S=Z(I,J)
      Z(I,J)=Z(K,J)
      Z(K,J)=S
40      CONTINUE
50      CONTINUE
60      RETURN
END

```



```

C-----DOUBLE QR STEP INVOLVING ROWS L TO EN AND COLUMNS M TO EN-----
  DO 190 K=M,NA
  NOTLAS=K.NE.NA
  IF (K.EQ.M) GO TO 120
  P=H(K,K-1)
  Q=H(K+1,K-1)
  R=C.0D0
  IF (NOTLAS) R=E(K+2,K-1)
  X=DABS(P)+DABS(Q)+DABS(R)
  IF (X.EQ.0.0D0) GO TO 190
  P=P/X
  Q=C/X
  R=S/X
  120 S=DSIGN(DSQRT(P*P+Q*C+R*R),P)
  IF (K.EQ.1) GO TO 130
  H(K,K-1)=-S*R
  GO TO 140
  130 IF (L.NE.M) H(K,K-1)=-H(K,K-1)
  140 P=P+S
  X=E/S
  Y=Q/S
  ZZ=R/S
  Q=Q/P
  R=R/P
C-----ROW MODIFICATION-----
  DO 160 J=K,EN
  P=H(K,J)+Q*H(K+1,J)
  IF (NOTLAS) GO TO 150
  P=P+R*H(K+2,J)
  H(K+2,J)=H(K+2,J)-P*ZZ
  150 H(K+1,J)=H(K+1,J)-P*Y
  H(K,J)=H(K,J)-E*X
  160 CONTINUE
  J=MIN0(EN,K+3)
C-----COLUMN MODIFICATION-----
  DO 180 I=L,J
  P=X*H(I,K)+Y*H(I,K+1)
  IF (NOTLAS) GO TO 170
  P=P+ZZ*X*H(I,K+2)
  H(I,K+2)=H(I,K+2)-P*Z
  H(I,K+1)=H(I,K+1)-P*Q
  H(I,K)=H(I,K)-E*Y
  180 CONTINUE
  190 CONTINUE
  GO TO 40
C-----ONE ROOT FOUND-----
  200 WR(EN)=X+T
  WI(EN)=0.0D0
  EN=NA
  GO TO 30
C-----TWO ROOTS FOUND-----
  210 P=(Y-X)/2.0D0
  Q=P*P+R
  ZZ=DSQRT(DABS(Q))
  X=X+T
  IF (Q.LT.0.0D0) GO TO 220
C-----REAL PAIR-----
  ZZ=P+DSIGN(ZZ,P)
  WR(NA)=X+ZZ
  WI(NA)=WE(NA)
  IF (ZZ.NE.0.0D0) WR(EN)=X-W/ZZ
  WI(NA)=0.0D0
  WI(EN)=0.0D0
  GO TO 230
C-----COMPLEX PAIR-----
  220 WE(NA)=X+P
  WR(EN)=X+P
  WI(NA)=ZZ
  WI(EN)=-ZZ
  230 EN=ENM2
  GO TO 30
C-----SET ERROR -- NO CONVERGENCE TO AN-----
C-----EIGENVALUE AFTER 30 ITERATIONS-----
  240 IERR=EN
  250 RETURN
  END

```



```

C=====
C      SUBROUTINE PSDCAL (N2,NS,FA,X,NC,GN,GV,C,NO,HY,HU,H,
1      FBGE,NG,GAM,ACL,F,WR,W1,D1,D2,JCF,RES,Q,R,BB,CC,IYU,
2      IPSD,INORM)
C=====
C      = PSDCAL COMPUTES THE PSD OF OUTPUTS OF CONTROLS OF =
C      = A CONTROLLED SYSTEM                                     =
C
C      =   IYU= 1      CUTPUT PSD                                =
C      =   = 2      CCNTROL ESD                                =
C      =   = 3      EOTH OUTPUT AND CONTROL PSD                 =
C
C      =   IPSD=1      PSD                                     =
C      =   =2      PSD AND TF RESIDUES                         =
C
C      =   INCRM=      1,2... NG  NORMALIZED BY ITH PROCESS NOISE =
C      =   NG+1... NG+NC NORMALIZED BY ITH MEAS NOISE          =
C=====
C      DOUBLE PRECISION FA,X,GN,GV,C,HY,H,FBGE,GAM,ACL,F,WR,W1,D1,D2,RES,
1      BB,CC,Q,B,PSD,W,INORM,DN1,EMAX,ELOG,EMOD,DW,ST,OM,RE,AI,HU,DW1
C      COMPLX2 X*16ZD ZN,2Z
C      DIMENSION FA(N2,N2),X(N2,N2),GN(N2,NG),C(NC,NS),HY(NO,N2),H(NC,NS)
1      FBGE(NS,NO),GAM(NS,NG),ACL(NS,NS),F(NS,NS),WR(N2),W1(N2),D1(N2),D
2      (N2),RES(N2),C(NG,NG),R(NO,NC),PSD(30),W(30),BB(N2),CC(N2),GV(N2,
3      NC),HU(NC,N2),DW1(4)
C      INTEGER JCF(N2)
C      DATA DW1/1.00,2.00,5.00,10.00/
C      IF (IYU.EQ. 0) IYU=1
C      IF (INORM.EQ. 0) INORM=1
C      IPT=0
C      IF (IPSD .GT. 1) IPT=1
C      IX=INORM-NG
C      IF (IX .GT. 0) WRITE (6,330) IX
C      IF (IX .LE. 0) WRITE (6,340) INCRM
C      NSC=N2*N2
C=====COMPUTE EIGENSYSTEM OF CONTROLLED SYSTEM; FORM FA-----
C      DC 10 I=1,NS
10     DO 10 J=1,NS
C      FA(I,J)=ACL(I,J)
C      FA(NS+I,J)=0.00
C      DO 30 I=1,NS
C      DO 30 J=1,NS
C      ST=0.00
C      DO 20 K=1,NO
20     ST=ST+PBGE(I,K)*H(K,J)
C      FA(I,NS+J)=-ST
30     FA(NS+I,NS+J)=F(I,J)-ST
C      CALL RAPENT (N2,N2,N2,9,FA,4,'(9(1X,1PD13.6))')
C=====DEBUG ABOVE
C      CALL BALANC (N2,N2,FA,LOW,IHIGH,D1)
C      CALL ORTHES (N2,N2,LCW,IHIGH,FA,D2)
C      CALL ORTEAN (N2,N2,LOW,IHIGH,FA,D2,X)
C      CALL HQR2 (N2,N2,LOW,IHIGH,FA,WR,W1,X,IERR)
C      IF (IERR .NE. 0) GO TO 320
C      CALL BALBAK (N2,N2,LCW,IHIGH,D1,N2,X)
C      CALL RAPENT (N2,N2,N2,9,X,4,'(9(1X,1PD13.6))')
C=====DEBUG ABCVE; DETERMINE MCDAL MATRICES
C      IF (IYU .EQ. 1) GC TO 60
C=====HSUBU
C      DO 50 I=1,NC
C      DO 50 J=1,N2
C      ST=0.00
C      DC 40 K=1,NS
40     ST=ST-C(I,K)*Y(F,J)
C      HU(I,J)=ST
C      GO TO 90
C=====HSUBY
60     DO 80 I=1,NO
C      DO 80 J=1,N2
C      ST=0.00
C      DC 70 K=1,NS
70     ST=ST+H(I,K)*X(F,J)-H(I,K)*X(NS+K,J)
C      HY(I,J)=ST
C      CALL RAPRNT (NC,NC,N2,9,HY,4,'(9(1X,1PD13.6))')
C=====DEBUG AEOVE

```



```

90      CALL MINV (NSQ,X,NZ,ST,D1,D2)
      CALL RAPRNT (N2,N2,N2,9,X,4,'(9(1X,1ED13.6))')
C----- DEBUG ABOVE -----
C----- GSUB W -----
C----- DO 110 I=1,N2
      DO 110 J=1,NG
      ST=0.0 DO
      DO 100 K=1,NS
100      ST=ST-X(I,NS+K)*GAM(K,J)
      GW(I,J)=ST
      CALL RAPRNT (N2,N2,NG,9,GW,4,'(9(1X,1ED13.6))')
C----- DEBUG ABOVE : USE SELECTED NORMALIZATION -----
      IF (INORM .LE. NG) DNORM=1. DO/C(INORM,INORM)
      IF (INORM .GT. NG) DNORM=1. DO/R(INORM-NG,INORM-NG)
C----- DETERMINE BANDWIDTH OF CONTROLLED SYSTEM -----
      EMAX=0. DO
      DO 120 I=1,N2
      EMOD=DABS(WR(I)**2 + WI(I)**2)
      IF (EMOD .GT. EMAX) EMAX=EMOD
120      CONTINUE
      EMOD=DSORT(EMAX)
      EMOD=2*EMOD
C----- ROUND UP TO NEAREST 2,4,5,8,10 -----
      ELOG=DLOG10(EMOD)
      IF (ELOG .LT. 0. DC) IPOW=-IDINT(DAES(ELOG) + 1)
      IF (ELOG .GE. 0. DC) IPOW=IDINT(ELOG)
      EMAX=EMOD*10**(-IPOW)
      IF (EMAX .GT. 2. DO) EMOD=2. DO
      IF (EMAX .GT. 4. DO) EMOD=4. DO
      IF (EMAX .GT. 5. DO) EMOD=5. DO
      IF (EMAX .GT. 8. DO) EMOD=8. DO
      IF (EMAX .GE. 10. DO) EMOD=10. DO
      EMAX=EMOD*10**IECW
      DW=EMAX/20. DO
C----- ADD 10 POINTS 3 DECADES UP -----
      IF (EMOD .LT. 5.0) GC TO 130
      EMAX=1.0D1
      IK=3
      GC TO 140
130      EMAX=5. DO
      IK=2
140      CONTINUE
C----- STORE 30 FREQUENCIES -----
      DC 150 I=1,20
150      W(I)=DW*(I-1)
      DC 160 I=1,3
      IP=20+3*(I-1)
      DO 160 J=1,3
      IX=MOD(IK+J-1,3)+1
      JJ=0
      IF (IK .EQ. 2 .AND. J .GE. 2) JJ=1
      W(IP+J)=DW1(IX)*10**((IPOW+I-1+JJ+IK-2))
160      CONTINUE
      IX=MOD(IK,3)+1
      W(30)=DW1(IX)*10**((IPOW+3+IK-2))
C----- LARGE LOOP THRU OUTPUTS -----
      IF (IYU .EQ. 1) NI=NC
      IF (IYU .EQ. 2) NI=NC
      DO 310 L=1,NL
      DO 170 I=1,30
      PSD(I)=0. DO
C----- LOOP THRU PROCESS NOISE -----
      DO 220 I=1,NG
      DN1=DNORM*0(I,I)
      IF (IYU .EQ. 1 .AND. IPT .EQ. 1) WRITE (6,350) I,L
      IF (IYU .EQ. 2 .AND. IPT .EQ. 1) WRITE (6,360) I,L
      IF (IYU .EQ. 1) CALL RESID (I,L,N2,JCF,NG,GW,NL,HY,WR,WI,
      1RES,BB,CC,IPT)
      IF (IYU .EQ. 2) CALL RESID (I,L,N2,JCF,NG,GW,NL,HU,WR,WI,
      1RES,BB,CC,IPT)
      DO 210 K=1,20
      ZZ=DCMPLX(0.0D0,0.0D0)
      OM=W(K)
      DO 200 II=1,N2
      IF (WI(II)) 200, 180, 190
180      ZD=DCMPLX(-WR(II),CM-WI(II))

```



```

190      ZZ=RES(II)/ZD+ZZ
      GC TO 200
      RE=WR(II)
      AI=WI(II)
      ZD=DCMPLX( RE**2 + AI**2 - OM**2, -2.D0*RE*OM)
      ZN=DCMPLX( RES(II+1)*AI-RES(II)*RE, RES(II)*OM)
      ZZ=ZZ+ZN/ZD
200      CCCONTINUE
210      PSD(K)=PSD(K)+DN1*(ZZ*DCONJG(ZZ))
220      CCCONTINUE
C-----GSUBV-----
      DO 240 I=1,N2
      DO 240 J=1,NO
      ST=0.D0
      DO 230 K=1,NS
      ST=ST+K(I,K)*FEGE(K,J)+X(I,NS+K)*FEGE(K,J)
      GV(I,J)=ST
      CALL RAPENT(N2,N2,NO,9,GV,4,'(9(1X,1PD13.6))')
C-----DEBUG ABCVE, LOOP THRU MEAS NOISE-----
      DO 300 I=1,NO
      DN1=DNORM*R(I,I)
      IF (IYU.EQ.1) AND (IPT.EQ.1) WRITE(6,370) I,L
      IF (IYU.EQ.2) AND (IPT.EQ.1) WRITE(6,380) I,L
      IF (IYU.EQ.1) CALL RESID(I,L,N2,JCF,NO,GV,NL,HY,WR,NI,RES,
      1 BB,CC,IPT)
      IF (IYU.EQ.2) CALL RESID(I,L,N2,JCF,NO,GV,NL,HU,WR,NI,RES,
      1 BB,CC,IPT)
      DO 290 K=1,30
      ZZ=DCMPLX(0.DC,0.I0)
      OM=W(K)
      DC 270 I=1,N2
      IF (WI(II)) 270,250,260
250      ZD=DCMPLX(-WR(II),OM-WI(II))
      ZZ=ZZ+RES(II)/ZD
      GC TO 270
260      RE=WR(II)
      AI=WI(II)
      ZD=DCMPLX( RE**2 + AI**2 - CM**2, -2.D0*RE*OM)
      ZN=DCMPLX( RES(II+1)*AI-RES(II)*RE, RES(II)*OM)
      ZZ=ZZ+ZN/ZD
270      CCCONTINUE
      IF (IYU.EQ.2) CB. I .NE. L GO TO 280
      PSD(K)=PSD(K)+DN1*(ZZ*DCONJG(ZZ))
280      PSD(K)=PSD(K)+DN1*(ZZ*DCONJG(ZZ))
290      CCCONTINUE
300      CCCONTINUE
      IF (IYU.EQ.1) WRITE(6,390) I
      IF (IYU.EQ.2) WRITE(6,400) I
      WRITE(6,410) (W(I),PSD(I),I=1,30)
310      CCCONTINUE
      RETURN
320      CCCONTINUE
      CALL BEXIT(N2,FA,IERR)
      RETURN
C-----FORMAT -----
330      FORMAT(//,41H SUBSEQUENT PSD IS NORMALIZED BY MEAS NO.,I3)
340      FORMAT(//,50H SUBSEQUENT PSD IS NORMALIZED BY PROCESS NOISE NO.,I3
      1)
350      FORMAT(/,38H TRANSFER FUNCTION FROM PROCESS NOISE ,I2,3H TO,13H ME
      1ASUREMENT,I2)
360      FORMAT(/,38H TRANSFER FUNCTION FROM PROCESS NOISE ,I2,3H TO,9H CCN
      1TRCL,I2)
370      FORMAT(/,36H TRANSFER FUNCTION FROM MEASUREMENT ,I2,16H TO MEASURE
      1MENT,I2)
380      FORMAT(/,36H TRANSFER FUNCTION FROM MEASUREMENT ,I2,12H TO CONTROL
      1,I2)
390      FORMAT(/,14H PSD CF CUTPUT,I3,32H FORCED BY ALL NOISE-(RAD FREQ.,
      115HNORMALIZED PSD)/)
400      FORMAT(/,15H PSD CF CCNTRCL,I3,32H FORCED BY ALL NOISE-(RAD FREQ,
      1,15HNORMALIZED PSD)/)
410      FORMAT(4(1X,1H(,E11.4,1H,,E11.4,1H)))
      END

```



```
C=====
C      SUBROUTINE ERExit (N,A,IERR)
C      ERExit RETURNS THE NUMBER OF THE EIGENVALUE WHERE HQR2      =
C      FAILS, THEN STOPS THE PROGRAM.                                =
C=====
C      INTEGER IERR
C      DOUBLE PRECISION A
C      DIMENSION A(N,N)
C      WRITE (5,10) IERR
C      CALL RAPANT (N,N,N,9,A,4,'(9(1X,1PD13.6))')
C      RETURN
10     FORMAT (35H FAILURE IN HQR2 ON EIGENVALUE NO. ,I3)
C      END
```



```

C=====
C      SUBROUTINE READF (NS,ISAF,BA)
C      INTERACTIVELY ENTERS THE "F" MATRIX ELEMENT BY ELEMENT.
C=====
C
      REAL*8  BA(NS,NS),DUM,ANSR
      INTEGER  I,J,K,L,IANS,ISAF
      DATA IY/'Y','/IZ/'N','/
      IF (ISAF.EQ.1) GO TO 40
      WRITE (5,130)
      DO 20 I=1,NS
      DO 10 J=1,NS
      WRITE (5,120) I,J
      CALL RDREAL (ANSR)
      BA(I,J)=ANSR
10    CONTINUE
20    CONTINUE
C-----
30    CALL PRTCMS ('CIRSCBN ')
40    CONTINUE
      WRITE (5,140)
      CALL MATERT (EA,NS,NS)
50    WRITE (5,150)
      CALL RDCHAR (IANS)
      IF ((IANS.NE.IY).AND.(IANS.NE.IZ)) GC TC 60
      GO TO 70
60    WRITE (5,160)
      GO TO 50
70    CONTINUE
      IF (IANS.EQ.IZ) GC TO 110
      IF (IANS.EQ.IY) GC TC 80
80    WRITE (5,170)
      CALL RDINT (IANS)
      K=IANS
      WRITE (5,180)
      CALL RDINT (IANS)
      L=IANS
      WRITE (5,120) K,L
      CALL RDREAL (ANSR)
      DUM=ANSR
      DO 100 I=1,NS
      DO 90 J=1,NS
      IF ((I.EQ.K).AND.(J.EQ.L)) BA(I,J)=DUM
90    CONTINUE
100   CONTINUE
      GO TO 30
110   CONTINUE
      CALL PRTCMS ('CLRSCKN ')
      RETURN
C-----
120  FORMAT (5X,14HTHE ELEMENT F{,I2,1H,I2,2H}|)
130  FORMAT (/,5X,36HENTER THE SYSTEM MATRIX "F"-MATRIX ,//,10X,41HDIM
140  TENSION = # STATES NS X # STATES NS )
150  FORMAT (//,15X,33HTHE SYSTEM MATRIX "F"-MATRIX
160  1ENT? //,10X,19HTYPEE "YES" OR "NO".)
170  FORMAT (1X,51HWARNING: IMPROPER DATA ENTRY! ENTER "YES" OR "NO".)
180  FORMAT (5X,50HENTER THE ROW NUMBER OF THE ELEMENT TO BE CHANGED.)
190  FORMAT (5X,53HENTER THE COLUMN NUMBER OF THE ELEMENT TO BE CHANGED
1.)
      END

```



```

C=====
C      SUBROUTINE READH (NC,NS,ISAH,HC)
C      INTERACTIVELY ENTERS THE "H" MATRIX MEASUREMENT SCALING MATRIX .=
C=====
C      REAL*8  HC(NO,NS),DUM,ANSR
C      INTEGER IANS,I,J,I,L,ISAH
C      DATA IY/'Y'/,IZ/'N'/
C
C-----TEIS IS AN EXAMPLE OF ONE POSSIBLE METHOD OF ARRAY GENERATION =
C-----WITHIN THE PROGRAM ITSELF. FOR VERY LARGE DATA ARRAYS, THIS METHOD =
C-----MAY BE PREFERABLE TO SOME USERS OVER INTERACTIVE ENTRY OF EACH =
C-----INDIVIDUAL ELEMENT.
C-----DO 2 I=1,11
C-----  DO 1 J=1,82
C-----    HC(I,J) = 0.0D+00
C-----    HC(1,1) = 0.11520D+00
C-----    HC(2,75) = 0.5730D+02
C-----    HC(3,74) = 0.1000D+01
C-----    HC(4,63) = 0.5730D+02
C-----    HC(5,62) = 0.1000D+01
C-----    HC(6,76) = 0.5730D+02
C-----    HC(7,44) = 0.5730D+02
C-----    HC(8,45) = 0.5730D+02
C-----    HC(9,46) = 0.5730D+02
C-----    HC(10,47) = 0.5730D+02
C-----    HC(11,48) = 0.5730D+02
C1      CONTINUE
C2      CONTINUE
C      GO TO 90
C3      CCNTINUE
C-----IF (ISAH.EQ.1) GO TO 40
C-----WRITE (5,120)
C-----DO 20 I=1,NO
C-----  DO 10 J=1,NS
C-----    WRITE (5,110) I,J
C-----    CALL RDREAL (ANSR)
C-----    HO(I,J)=ANSR
C10     CCNTINUE
C20     CCNTINUE
C-----CALL PRTCMS ('CLRSCFN ')
C-----CONTINUE
C-----WRITE (5,130)
C-----CALL MATPRT (HC,NC,NS)
C-----WRITE (5,140)
C-----CALL RDCHAR (IANS)
C-----IF ((IANS.NE.IY).AND.(IANS.NE.IZ)) GO TO 60
C-----GO TO 70
C-----WRITE (5,150)
C-----GO TO 50
C-----CCNTINUE
C-----IF (IANS.EQ.IZ) GC TC 100
C-----WRITE (5,160)
C-----CALL RDINT (IANS)
C-----K=IANS
C-----WRITE (5,170)
C-----CALL RDINT (IANS)
C-----L=IANS
C-----WRITE (5,110) K,L
C-----CALL RDREAL (ANSR)
C-----DUM=ANSR
C-----DO 90 I=1,NO
C-----  DO 80 J=1,NS
C-----    IF ((I.EQ.K).AND.(J.EQ.L)) HO(I,J)=DUM
C80     CCNTINUE
C90     CCNTINUE
C-----GC TO 30
C100    CCNTINUE
C-----CALL PRTCMS ('CLRSCRN ')
C-----RETURN
C-----110  FORMAT (5X,14HTHE ELEMENT H (,I2,1H,,I2,2H)=)
C120  FORMAT (/5X,5CENTER THE MEASUREMENT SCALING MATRIX "H"-MATRIX .
C-----1,/,10X,4HDIMENSION = # OBSERVATIONS NO X # STATES NS )

```



```
130  FORMAT (//,10X,46HTHE MEASUREMENT SCALING MATRIX "H"-MATRIX ...,/1/)  
140  FORMAT (//5X,54HDC YOU WISH TO CHANGE THE VALUE OF ANY MATRIX ELEMENT?//,10X,19HTYPEF "YES" OR "NO".)  
150  FORMAT (1X,51HWARNING: IMPROPER DATA ENTRY! ENTER "YES" OR "NO".)  
160  FORMAT (5X,50HENTER THE ROW NUMBER OF THE ELEMENT TO BE CHANGED.)  
170  FORMAT (5X,52HENTER THE COLUMN NUMBER OF THE ELEMENT TO BE CHANGED)  
1}  
END
```



```

C=====
C      SUBROUTINE READD (NC, NC, D)
C      ENTERS THE "D" MATRIX MEASUREMENT FEED-FORWARD DIST. MATRIX.
C=====
C
      REAL*8  D (NO, NC), DUM, ANSR
      INTEGER  IANS, I, J, K, L
      DATA  IY /'Y'/, IZ /'N'/
      WRITE (5, 110)
      DO 20 I=1, NO
      DO 10 J=1, NC
      WRITE (5, 100)  I, J
      CALL RDREAL (ANSR)
      D (I, J) = ANSR
10    CONTINUE
20    CONTINUE
C-----
      CALL FRTCMS ('CLRSCRN ')
      WRITE (5, 120)
      CALL MATPBT (D, NC, NC)
      WRITE (5, 130)
      CALL RDCHAR (IANS)
      IF ((IANS.NE.IY).AND.(IANS.NE.IZ))  GC TO 50
      GC TO 60
50    WRITE (5, 140)
      GO TO 40
60    CONTINUE
      IF (IANS.EQ.IZ)  GC TO 90
      WRITE (5, 150)
      CALL RDINT (IANS)
      K=IANS
      WRITE (5, 160)
      CALL RDINT (IANS)
      L=IANS
      WRITE (5, 100)  K, L
      CALL RDREAL (ANSR)
      DUM=ANSR
      DC 80 I=1, NO
      DO 70 J=1, NC
      IF ((I.EQ.K).AND.(J.EQ.L))  D (I, J) = DUM
70    CONTINUE
80    CONTINUE
      GO TO 30
90    CONTINUE
      CALL FRTCMS ('CLRSCEN ')
      RETURN
C-----
100   FORMAT (5X, 14HENTER ELEMENT D ( IZ, 1H, IZ, 2H ) =)
110   FORMAT (/, 5X, 54HENTER THE MEASUREMENT FEEDTHROUGH MATRIX / FEEDFOR
120   FORMAT (//, 5X, 34H DISTRIBUTION MATRIX "D"-MATRIX . . . , 8X, 49H DIMENSION
2 = # OBSERVATIONS NC X # CONTROLS NC )
130   FORMAT (//, 5X, 54H THE FEEDFORWARD DISTRIBUTION MATRIX "D"-MATRIX .
1 . . . )
140   FORMAT (//, 10X, 19H TYPE "YES" OR "NO".)
150   FORMAT (1X, 51H WARNING: IMPROPER DATA ENTRY! ENTER "YES" OR "NO".)
160   FORMAT (5X, 50H ENTER THE ROW NUMBER OF THE ELEMENT TO BE CHANGED.)
   FORMAT (5X, 53H ENTER THE COLUMN NUMBER OF THE ELEMENT TO BE CHANGED
1 . . .
END

```



```

C=====
C      SUBROUTINE READEG (NS,NC,ISAG,G)
C      INTERACTIVELY ENTERS THE "G" MATRIX  CONTROL DISTRIBUTION MATRIX =
C=====
C
      REAL*8 G(NS,NC),DUM,ANSR
      INTEGER IANS,I,J,K,I,ISAG
      DATA IY/'Y'/,IZ/'N'/
      IF (ISAG.EQ.1) GO TO 40
      WRITE (5,120)
      DO 20 I=1,NS
      DO 10 J=1,NC
      WRITE (5,110) I,J
      CALL RDREAL (ANSR)
      G(I,J)=ANSR
10    CONTINUE
20    CCNTINUE
C-----
30    CALL FETCMS ('CLRSCRN ')
40    CCNTINUE
      WRITE (5,130)
      CALL MATPRT (G,NS,NC)
50    WRITE (5,140)
      CALL RDCHAR (IANS)
      IF ((IANS.NE.IY).AND.(IANS.NE.IZ)) GO TO 60
      GO TO 70
60    WRITE (5,150)
      GO TO 50
70    CCNTINUE
      IF (IANS.EQ.IZ) GO TO 100
      WRITE (5,160)
      CALL RDINT (IANS)
      K=IANS
      WRITE (5,170)
      CALL RDINT (IANS)
      L=IANS
      WRITE (5,110) K,L
      CALL RDREAL (ANSR)
      DUM=ANSR
      DO 90 I=1,NS
      DO 80 J=1,NC
      IF ((I.EQ.K).AND.(J.EQ.L)) G(I,J)=DUM
80    CCNTINUE
90    CCNTINUE
      GO TO 30
100   CCNTINUE
      CALL FETCMS ('CLRSCRN ')
      RETURN
C-----
110   FORMAT (5X,14HTHE ELEMENT G(,I2,1H,,I2,2H)=)
120   FORMAT (/,5X,51HENTER THE CONTROL DISTRIBUTION MATRIX "G"-MATRIX
130   1: //,10X,43HDIMENSION = # STATES NS X # CONTROLS NC )
      FORMAT (//,10X,47HTHE CONTROL DISTRIBUTION MATRIX "G"-MATRIX ...,
140   1//)
      FORMAT (//5X,54HDC YOU WISH TO CHANGE THE VALUE OF ANY MATRIX ELEM
      ENT? //,1CX,19HTYPEE "YES" OR "NO".)
150   FORMAT (1X,51HWARNING: IMPROPER DATA ENTRY! ENTER "YES" OR "NO".)
160   FORMAT (5X,51HENTER THE ROW NUMBER OF THE ELEMENT TO BE CHANGED.)
170   FORMAT (5X,53HENTER THE COLUMN NUMBER OF THE ELEMENT TO BE CHANGED
1: )
      END

```



```

C=====
C      SUBROUTINE READFB (NC,NS,FBGC)
C      ENTERS THE "C" MATRIX  FEEDBACK GAIN CONTROL MATRIX .
C=====
C      REAL*8  FEGC (NC,NS), DUM,ANSR
C      INTEGER IANS,I,J,K,L
C      DATA IY/'Y',IZ/'N'
C      WRITE (5,110)
C      DC 20 I=1,NC
C      DC 10 J=1,NS
C      WRITE (5,100) I,J
C      CALL RDREAL (ANSR)
C      FBGC (I,J)=ANSR
10    CONTINUE
20    CONTINUE
C-----
30    CALL PRTCMS ('CLRSCN ')
C      WRITE (5,120)
C      CALL MATPRT (FEGC,NC,NS)
40    WRITE (5,130)
C      CALL RDCHAR (IANS)
C      IF ((IANS.NE.IY).AND.(IANS.NE.IZ)) GC TO 50
C      GC TO 60
50    WRITE (5,140)
C      GC TO 40
60    CONTINUE
C      IF (IANS.EQ.IZ) GC TO 90
C      WRITE (5,150)
C      CALL RDINT (IANS)
K=IANS
C      WRITE (5,160)
C      CALL RDINT (IANS)
L=IANS
C      WRITE (5,100) K,L
C      CALL RDREAL (ANSR)
DUM=ANSR
DC 80 I=1,NC
DO 70 J=1,NS
C      IF ((I.EQ.K).AND.(J.EQ.L)) FBGC (I,J) =DUM
70    CONTINUE
80    CONTINUE
GO TO 30
90    CONTINUE
CALL PRTCMS ('CLRSCRN ')
RETURN
C-----
100   FFORMAT (5X,14HENTER ELEMENT C(I,I2,1H,I2,2H)=)
110   FORMAT (/,5X,52HENTER THE FEEDBACK GAIN CONTROL MATRIX "C"-MATRIX
1      //,10X,44HDIMENSION = * CONTROLS NC X * STATES NS *)
120   FORMAT (/,10X,45HTHE FEEDBACK GAIN CCNTROL MATRIX "C"-MATRIX ,//)
130   FORMAT (//5X,54HDC YOU WISH TO CHANGE THE VALUE OF ANY MATRIX ELEM
1      ENT?//,1CX,19HTYPE "YES" OR "NC".)
140   FFORMAT (1X,51HWARNING: IMPROPER DATA ENTRY! ENTER "YES" OR "NO".)
150   FORMAT (5X,50HENTER THE ROW NUMBER OF THE ELEMENT TO BE CHANGED.)
160   FFORMAT (5X,53HENTER THE COLUMN NUMBER OF THE ELEMENT TO BE CHANGED
1      )
END

```



```

C=====
C      SUBROUTINE REAAY (NC,AY)
C      ENTERS THE "A" MATRIX  DIAGONAL OUTPUT COST MATRIX .
C=====
C      REAL*8   AY(NO,NC),DUM,ANSR
C      INTEGER 3,IANS,I,J,L
C      DATA IY/'Y',IZ/'N'/
C      WRITE (5,110)
C      DC 20 I=1,NO
C      DC 10 J=1,NO
C      WRITE (5,100) I,J
C      CALL RDREAL (ANSR)
C      AY(I,J)=ANSR
10    CONTINUE
20    CONTINUE
C-----
30    CALL FRTCMS ('CLRSCEN ')
C      WRITE (5,120)
C      CALL MATPFT (AY,NC,NC)
40    WRITE (5,130)
C      CALL RDCHAR (IANS)
C      IF ((IANS.NE.IY).AND.(IANS.NE.IZ)) GC TO 50
C      GO TO 60
50    WRITE (5,140)
C      GO TO 40
60    CONTINUE
C      IF (IANS.EQ.IZ) GC TO 90
C      WRITE (5,150)
C      CALL RDINT (IANS)
K=IANS
C      WRITE (5,160)
C      CALL RDINI (IANS)
L=IANS
C      WRITE (5,100) K,L
C      CALL RDREAL (ANSR)
DUM=ANSR
DC 80 I=1,NO
DO 70 J=1,NO
C      IF ((I.EQ.K).AND.(J.EQ.L)) AY(I,J)=DUM
70    CONTINUE
80    CCNTINUE
C      GO TO 30
90    CCNTINUE
C      CALL FRTCMS ('CLRSCEN ')
C      RETURN
C-----
100   FC5MAT (5X,14HTYPE ELEMENT A(I2,I1H,I2,2H)=)
110   FORMAT (//,5X,54HENTER THE OUTPUT MEASUREMENT COST MATRIX "A"-MAT
1      RIX .,.,5X,53HDIMENSION = * OBSERVATIONS NO X * OBSERVATIONS NO
2      )
120   FORMAT (//,5X,50HTHE OUTPUT MEASUREMENT COST MATRIX "A"-MATRIX ..
1      .)
130   FC5MAT (//,5X,54HDC YOU WISH TO CHANGE THE VALUE OF ANY MATRIX ELEM
1      ENT? //,1CX,19HTYPE "YES" OR "NO".)
140   FORMAT (1X,51HWARNING: IMPROPER DATA ENTRY! ENTER "YES" OR "NO".)
150   FORMAT (5X,50HENTER THE ROW NUMBER OF THE ELEMENT TO BE CHANGED.)
160   FORMAT (5X,53HENTER THE COLUMN NUMBER OF THE ELEMENT TO BE CHANGED
1      .)
C      END

```



```

C=====
C      SUBROUTINE READS (NC,B)
C      ENTERS THE "B" MATRIX CONTROL COST WEIGHTING MATRIX .
C=====
C      REAL*B8 E(NC,NC),DUM,ANSR
C      INTEGER IANS,I,J,K,L
C      DATA IY/'Y'/,IZ/'N'/
C      WRITE (5,90)
C      DC 10 I=1,NC
C      DO 10 J=1,NC
C      WRITE (5,80) I,J
C      CALL RDREAL (ANSR)
10    B(I,J)=ANSR
C-----
C      20  CALL PRTCMS ('CLRSRN ')
C      WRITE (5,120)
C      CALL MATFRT (B,NC,NC)
30    WRITE (5,110)
C      CALL RDCHAR (IANS)
C      IF ((IANS.NE.IY).AND.(IANS.NE.IZ)) GC TO 40
C      GC TO 50
40    WRITE (5,120)
C      GC TO 30
50    CONTINUE
C      IF (IANS.EQ.IZ) GC TO 70
C      WRITE (5,130)
C      CALL RDINT (IANS)
C      K=IANS
C      WRITE (5,140)
C      CALL RDINT (IANS)
C      L=IANS
C      WRITE (5,80) K,L
C      CALL RDREAL (ANSR)
C      DUM=ANSR
C      DC 60 I=1,NC
C      DO 60 J=1,NC
C      IF ((I.EQ.K).AND.(J.EQ.L)) B(I,J)=DUM
60    CONTINUE
C      GC TO 20
70    CONTINUE
C      CALL PRTCMS ('CLESCRN ')
C      RETURN
C-----
80    FORMAT (5X,14HTHE ELEMENT B(I2,1H,I2,2H)=)
90    FORMAT (5X,5HENTER THE CONTROL COST WEIGHTING MATRIX "B"-Matri
1X //,10X,4HDIMENSION = # CONTROL NC X # CONTROLS NC )
100   FORMAT (//,10X,37HTHE CONTROL COST MATRIX "B" //)
110   FORMAT (//,5X,54EDC YOU WISH TO CHANGE THE VALUE OF ANY MATRIX ELEM
1ENT? //,10X,19HTYPE "YES" OR "NO")
120   FORMAT (1X,51HWARNING: IMPROPER DATA ENTRY! ENTER "YES" OR "NO")
130   FORMAT (5X,50HENTER THE ROW NUMBER OF THE ELEMENT TO BE CHANGED.)
140   FORMAT (5X,52HENTER THE COLUMN NUMBER OF THE ELEMENT TO BE CHANGED
1)
END

```



```

C===== SUBROUTINE READG2 (NS,NG,IGAM,GAM)
C      ENTERS THE "GAM" MATRIX  PROCESS NOISE DISTRIBUTION MATRIX .      =
C=====
C      REAL*8 GAM(NS,NG),DUM,ANSR
C      INTEGER IANS,I,J,K,L,IGAM
C      DATA IY/'Y'/,IZ/N/
C      IF (IGAM.EQ.1) GO TO 40
C      WRITE (5,120)
C      DC 20 I=1,NS
C      DO 10 J=1,NG
C      WRITE (5,110) I,J
C      CALL RDREAL (ANSR)
C      GAM(I,J)=ANSR
10    CONTINUE
20    CCNTINUE
C-----
30    CALL PTCMS ('CLASCRN ')
40    CONTINUE
45    WRITE (5,130)
50    CALL MATPT (GAM,NS,NG)
55    WRITE (5,140)
60    CALL RDCHAR (IANS)
65    IF ((IANS.EQ.IY).AND.(IANS.NE.IZ)) GO TO 60
66    GO TO 70
67    WRITE (5,150)
68    GO TO 50
70    CONTINUE
75    IF (IANS.EQ.IZ) GO TO 100
76    WRITE (5,160)
77    CALL RDINT (IANS)
78    K=IANS
79    WRITE (5,170)
80    CALL RDINT (IANS)
81    L=IANS
82    WRITE (5,110) K,L
83    CALL RDREAL (ANSR)
84    DUM=ANSR
85    DO 90 I=1,NS
86    DO 80 J=1,NG
87    IF ((I.EQ.K).AND.(J.EQ.L)) GAM(I,J)=DUM
88    CONTINUE
89    CONTINUE
90    GO TO 30
100   CONTINUE
110   CALL PTCMS ('CLRSCRN ')
120   RETURN
C-----
110   FORMAT (5X,16H THE ELEMENT GAM(I2,1H,I2,2H) =)
120   FORMAT (/,5X,36H ENTER THE PROCESS NOISE DISTRIBUTION,/,5X,24H Matri
130   1X "GAMMA"-MATRIX .,/,2X,56HDIMENSION = * STATES NS * PROCESS
140   2NOISE SOURCES NG )
150   1"Gamma"-MATRIX //)
160   1FORMAT (/,5X,54HDC YOU WISH TO CHANGE THE VALUE OF ANY MATRIX ELEM
170   1ENT? //,1CX,19H TYPEEE "YES" OR "NO".)
180   1FORMAT (1X,51HWARNING: IMPROPER DATA ENTRY! ENTER "YES" OR "NO".)
190   1FORMAT (5X,50H ENTER THE ROW NUMBER OF THE ELEMENT TO BE CHANGED.)
200   1FORMAT (5X,53H ENTER THE COLUMN NUMBER OF THE ELEMENT TO BE CHANGED
1..)
END

```



```

C=====
C      SUBROUTINE READQ (NG,Q)
C      INTERACTIVELY ENTERS THE "Q" MATRIX NOISE WEIGHTING MATRIX      =
C=====
C
      REAL*8 C(NG,NG),DUM,ANSR
      INTEGER IANS,I,J,K,L
      DATA IY/'Y',IZ/'N'
      WRITE (5,110)
      DO 20 I=1,NG
      DO 10 J=1,NG
      WRITE (5,100) I,J
      CALL RDREAL (ANSR)
      Q(I,J)=ANSR
10    CONTINUE
20    CONTINUE
C=====
30    CALL FRTCMS ('CIRSC&N ')
      WRITE (5,120)
      CALL MATFRT (Q,NG,NG)
40    WRITE (5,130)
      CALL RDCHAR (IANS)
      IF ((IANS.NE.IY).AND.(IANS.NE.IZ)) GO TO 50
      GO TO 60
50    WRITE (5,140)
      GO TO 40
60    CONTINUE
      IF (IANS.EQ.IZ) GO TO 90
      WRITE (5,150)
      CALL RDINT (IANS)
      K=IANS
      WRITE (5,160)
      CALL RDINT (IANS)
      L=IANS
      WRITE (5,100) K,L
      CALL RDREAL (ANSR)
      DUM=ANSR
      DO 80 I=1,NG
      DO 70 J=1,NG
      IF ((I.EQ.K).AND.(J.EQ.L)) Q(I,J)=DUM
70    CONTINUE
80    CONTINUE
      GO TO 30
90    CONTINUE
      CALL FRTCMS ('CIRSC&N ')
      RETURN
C=====
100   FORMAT (5X,14HTYPE ELEMENT Q(,IZ,1H,,IZ,2H)=)
110   FORMAT (//,5X,44HENTER THE PROCESS NOISE PSD WEIGHTING MATRIX.,,5X
1,12H "0" MATRIX.,,5X,42H DIMENSION = # PROCESS NOISE SOURCES # NG
120   FORMAT (//,5X,27H #PROCESS NOISE SOURCES # NG)
130   FORMAT (//,5X,42HTHE PROCESS NOISE WEIGHTING MATRIX...Q...//)
140   FORMAT (//,5X,54HDO YOU WISH TO CHANGE THE VALUE OF ANY MATRIX ELEM
1ENT? //,10X,19HTYPE "YES" OR "NO".)
150   FORMAT (1X,51HWARNING: IMPROPER DATA ENTRY! ENTER "YES" OR "NO".)
160   FORMAT (5X,50HENTER THE ROW NUMBER OF THE ELEMENT TO BE CHANGED.)
170   FORMAT (5X,53HENTER THE COLUMN NUMBER OF THE ELEMENT TO BE CHANGED
180   END

```



```

C=====
C      SUBROUTINE READER (NO,RC)
C      ENTERS THE "R" MATRIX MEASUREMENT NOISE DISTRIBUTION MATRIX .  =
C=====
C      REAL*8  RC(NO,NO),DUM,ANSR
C      INTEGER IANS,I,J,K,L
C      DATA IY/5,100/
C      DC 10 I=1,NO
C      DC 10 J=1,NO
C      WRITE (5,80) I,J
C      CALL RDREAL (ANSR)
10    RC(I,J)=ANSR
C-----
20    CALL FRTCMS ('CLRSCBN ')
C      WRITE (5,100)
C      CALL MATPRT (RC,NC,NC)
30    WRITE (5,110)
C      CALL RDCHAR (IANS)
C      IF ((IANS.NE.IY).AND.(IANS.NE.IZ)) GC TC 40
C      GC TO 50
40    WRITE (5,120)
C      GC TO 30
50    CONTINUE
C      IF (IANS.EQ.IZ) GC TC 70
C      WRITE (5,130)
C      CALL RDINT (IANS)
K=IANS
C      WRITE (5,140)
C      CALL RDINT (IANS)
L=IANS
C      WRITE (5,80) K,L
C      CALL RDREAL (ANSR)
DUM=ANSR
DC 60 I=1,NO
DO 60 J=1,NO
60  IF ((I.EQ.K).AND.(J.EQ.L)) RC(I,J)=DUM
GO TO 20
70  CONTINUE
C      CALL FRTCMS ('CLRSCBN ')
C      RETURN
C-----
80  FCRMAT  (5X,14HTEEE ELEMENT R(I2,1H,I2,2H)=)
90  FORMAT (/,5X,6HENTER THE MEASUREMENT NOISE DISTRIBUTION MATRIX " )
100 FORMAT (1R" MATRIX ",/,,5X,53H DIMENSION = # OBSERVATIONS NO X # OBSERVATIO
2NS NO )
100 FORMAT (//,15X,5G4THE MEASUREMENT NOISE DISTRIBUTION MATRIX.....R.
110 FORMAT (//,5X,54HDC YOU WISH TO CHANGE THE VALUE OF ANY MATRIX ELEM
110 ENT?//,10X,19HTYPEE "YES" OR "NO".)
120 FORMAT (1X,51HWARNING: IMPROPER DATA ENTRY! ENTER "YES" OR "NO".)
130 FORMAT (5X,50HENTER THE ROW NUMBER OF THE ELEMENT TO BE CHANGED.)
140 FORMAT (5X,52HENTER THE COLUMN NUMBER OF THE ELEMENT TO BE CHANGED
1)
END

```



```

C=====
C      SUBROUTINE READFE (NS,NO,FBGE)
C      INTERACTIVELY ENTERS THE "K" FEEDBACK GAIN ESTIMATOR MATRIX =
C=====
C      REAL*8  FEGE(NS,NC), DUM,ANSR
C      INTEGER IANS, I,J,K,L
C      DATA IY/'Y',IZ/'N'
C      WRITE (5,110)
C      DO 20 I=1,NS
C      DO 10 J=1,NO
C      WRITE (5,100) I,J
C      CALL RDREAL (ANSR)
C      FBGE(I,J)=ANSR
10    CONTINUE
20    CONTINUE
C-----
30    CALL PRTCMS ('CLRSCE1')
C      WRITE (5,120)
C      CALL MATERT (FEGE,NS,NO)
40    WRITE (5,130)
C      CALL RDCLEAR (IANS)
C      IF ((IANS.NE.IY).AND.(IANS.NE.IZ)) GC TC 50
C      GC TO 60
50    WRITE (5,140)
C      GC TO 40
60    CONTINUE
C      IF (IANS.EQ.IZ) GC TC 90
C      WRITE (5,150)
C      CALL RDINT (IANS)
C      K=IANS
C      WRITE (5,160)
C      CALL RDINT (IANS)
C      L=IANS
C      WRITE (5,100) K,L
C      CALL RDREAL (ANSR)
C      DUM=ANSR
C      DO 30 I=1,NS
C      DO 20 J=1,NO
C      IF ((I.EQ.K).AND.(J.EQ.L)) FBGE(I,J)=DUM
70    CONTINUE
80    CONTINUE
80    GO TO 30
90    CONTINUE
C      CALL PRTCMS ('CLRSCE1')
C      RETURN
C-----
100   FORMAT (5X,14BTE ELEMENT K, I2,1H, I2,2H)=)
110   FORMAT (//,5X,54HENTER THE FEEDBACK GAIN ESTIMATOR MATRIX "K"-MTR
1IX //,18X,54HDIMENSION = # STATES NS X # OBSERVATIONS NO.)
120   FORMAT (///,15X,47HTHE FEEDBACK GAIN ESTIMATOR MATRIX "K"-MTRIX,
1//),
130   FORMAT (///,5X,54HDO YOU WISH TO CHANGE THE VALUE OF ANY MATRIX ELE
1MENT? //,10X,19HTYPE "YES" OR "NO".)
140   FORMAT (1X,5I1HWARNING: IMPROPER DATA ENTRY! ENTER "YES" OR "NO".)
150   FORMAT (5X,5I1HENTER THE ROW NUMBER OF THE ELEMENT TO BE CHANGED.)
160   FORMAT (5X,5I2HENTER THE COLUMN NUMBER OF THE ELEMENT TO BE CHANGED
1)
C      END

```



```

C=====
C      SUBROUTINE READW (NG,WR)
C      INTERACTIVELY ENTERS "W0"  STEADY DISTURBANCE VECTOR
C=====
C
C      SEAL*3  WF(NG),CUM,ANSR
C      INTEGER IANS,I,K
C      DATA IY/'Y',IZ/'N'
C      WRITE (5,100)
C      DC 10 I=1,NG
C      WRITE (5,90) I
C      CALL RDREAL (ANSR)
C      WH(I)=ANSR
10    CONTINUE
C-----
20    CALL FRTCMS ('CLRSCEH')
      WRITE (5,110)
      WRITE (5,90) (WR(I),I=1,NG)
30    WRITE (5,120)
      CALL RDCLAR (IANS)
      IF ((IANS.NE.IY).AND.(IANS.NE.IZ)) GC TO 40
      GC TO 50
40    WRITE (5,130)
      GC TO 30
50    CONTINUE
      IF (IANS.EQ.IZ) GC TO 70
      WRITE (5,140)
      CALL RDINT (IANS)
      K=IANS
      WRITE (5,80) K
      CALL RDREAL (ANSR)
      DUM=ANSR
      DO 60 I=1,NG
      IF (I.EQ.K) WH(I)=DUM
60    CONTINUE
      GC TO 20
70    CONTINUE
      CALL FSTCMS ('CLRSCEH')
      RETURN
C-----
80    FORMAT (5X,15HTHE ELEMENT W0 (,I2,2H) =)
90    FORMAT (F12.5)
100   FORMAT (/,5X,57HENTER THE STEADY DISTURBANCE VECTOR MATRIX "W0"-M
1      ATRIX ./,10X,44H DIMENSION = # PROCESS NOISE SOURCES NG X 1)
110   FORMAT (/,15X,53HTHE STEADY DISTURBANCE VECTOR MATRIX "W0"-Matri
1      X ./,15X,53HTHE STEADY DISTURBANCE VECTOR MATRIX "W0"-Matri
120   FORMAT (//5X,54HDC YOU WISH TO CHANGE THE VALUE OF ANY MATRIX ELEM
1      ENT? //,10X,51HTYPEE "YES" OR "NO".)
130   FORMAT (1X,51HWARNING: IMPROPER DATA ENTRY! ENTER "YES" OR "NO".)
140   FORMAT (5X,50HENIER THE RCW NUMBER OF THE ELEMENT TO BE CHANGED.)
      END

```



```

C=====
C  SUBROUTINE RDREAL -- INTERACTIVELY READS A REAL NUMBER REPLY      =
C  INTO A FORTRAN PROGRAM.  IF THE USER INADVERTENTLY ENTERS A NULL      =
C  STRING THE S/R ISSUES A WARNING AND ALLOWS A RECOVERY.                  =
C=====
C-----SUBROUTINE RDREAL (AANSR)
      REAL*8 AANSR
      INTEGER COUNT
C-----
      COUNT=0
10    CONTINUE
      COUNT=COUNT+1
      IF (COUNT.LT.3) GO TO 20
      WRITE (5,60)
      GO TO 40
20    CONTINUE
      READ (5,*,END=30,ERR=30) AANSR
      RETURN
30    REWIND 5
      WRITE (5,50)
      GO TO 10
40    CONTINUE
      STOP
C-----
50    FORMAT (1X,64HWARNING: NULL STRINGS ARE NOT ALLOWED, ENTER A NUME
      RICAL VALUE.)
60    FORMAT (////,2X,42HPROGRAM KILLED - TWO NULL STRINGS ENTERED! ,/)
      END

```



```

C=====
C SUBROUTINE RDINT -- INTERACTIVELY READS AN INTEGER REPLY =
C INTO A FORTRAN PROGRAM.  IF THE USER INADVERTENLY ENTERS AN IMPROPER =
C DATA CHARACTER THE S/R ISSUES A WARNING AND ALLOWS A RECOVERY.  =
C=====
SUBROUTINE RDINT (IANS)
INTEGER CCNT,IANS
C-----
COUNT=0
10  CONTINUE
  COUNT=COUNT+1
  IF (COUNT.LT.3) GO TO 20
  WRITE (5,60)
  GO TO 50
20  CONTINUE
  READ (5,*,END=40,ERR=40) IANS
  IF (IANS) 40,40,30
30  CONTINUE
  RETURN
40  REWIND 5
  WRITE (5,70)
  GO TO 10
50  CONTINUE
  STOP
C-----
60  FORMAT (//,5X,49HPROGRAM TERMINATION - TWO IMPROPER DATA ENTRIES!!
70  1) FORMAT (1X,56HWARNING: IMPROPER DATA ENTRY!  ENTER A POSITIVE INTE
1GER.)
  END

```



```

C=====
C  SUBROUTINE RDCHAR (IANS) -- INTERACTIVELY READS A CHARACTER STRING REPLY =
C  ('YES' OR 'NO') INTO A FORTRAN PROGRAM. IF THE USER INADVERTENTLY =
C  ENTERS A NULL STRING THE S/R ISSUES A WARNING AND ALLOWS A RECOVERY=
C=====
      SUBROUTINE RDCHAR (IANS)
      INTEGER COUNT, IANS
      DATA 1Y/'Y'/,1Z/'N'/
C-----
      COUNT=0
10   CONTINUE
      COUNT=COUNT+1
      IF (COUNT.LT.3) GO TO 20
      WRITE (5,60)
      GO TO 40
20   CONTINUE
     REWIND 5
      READ (5,70,END=30,ERR=30) IANS
      RETURN
30   REWIND 5
      WRITE (5,50)
      GO TO 10
40   CONTINUE
      STOP
C-----
50   FORMAT (1X,60HWARNING: NULL STRINGS ARE NOT ALLOWED, ENTER "YES"
      10E "NO".)
60   FORMAT (1X,47HPROGRAM TERMINATION - TWO NULL STRINGS ENTERED!)
70   FORMAT (A1)
      END

```



```

C=====C
C  SUBROUTINE MATPRT -- DISPLAYS A TWO-DIMENSIONAL ARRAY (16 COLS. MAX)=
C  IN VARIABLE SCREEN FORMAT FOR USER EASE IN ROW IDENTIFICATION.  =
C=====C
      SUBROUTINE MATPRT (PRTT, NROW, NCCL)
      IMPLICIT REAL*8 (A-B,0-2)
      DIMENSION PRTT (NROW,NCOL)
C-----
      IF (NCOL.EQ.0)  NCCL=1
      IF (NCOL.EQ.1)  WRITE (5, 10)  ((PRTT (I,J), J=1,NCOL), I=1,NROW)
      IF (NCOL.EQ.2)  WRITE (5, 20)  ((PRTT (I,J), J=1,NCOL), I=1,NROW)
      IF (NCOL.EQ.3)  WRITE (5, 30)  ((PRTT (I,J), J=1,NCOL), I=1,NROW)
      IF (NCOL.EQ.4)  WRITE (5, 40)  ((PRTT (I,J), J=1,NCOL), I=1,NROW)
      IF (NCOL.EQ.5)  WRITE (5, 50)  ((PRTT (I,J), J=1,NCOL), I=1,NROW)
      IF (NCOL.EQ.6)  WRITE (5, 60)  ((PRTT (I,J), J=1,NCOL), I=1,NROW)
      IF (NCOL.EQ.7)  WRITE (5, 70)  ((PRTT (I,J), J=1,NCOL), I=1,NROW)
      IF (NCOL.EQ.8)  WRITE (5, 80)  ((PRTT (I,J), J=1,NCOL), I=1,NROW)
      IF (NCOL.EQ.9)  WRITE (5, 90)  ((PRTT (I,J), J=1,NCOL), I=1,NROW)
      IF (NCOL.EQ.10) WRITE (5,100) ((PRTT (I,J), J=1,NCOL), I=1,NROW)
      IF (NCOL.EQ.11) WRITE (5,110) ((PRTT (I,J), J=1,NCOL), I=1,NROW)
      IF (NCOL.EQ.12) WRITE (5,120) ((PRTT (I,J), J=1,NCOL), I=1,NROW)
      IF (NCOL.EQ.13) WRITE (5,130) ((PRTT (I,J), J=1,NCOL), I=1,NROW)
      IF (NCOL.EQ.14) WRITE (5,140) ((PRTT (I,J), J=1,NCOL), I=1,NROW)
      IF (NCOL.EQ.15) WRITE (5,150) ((PRTT (I,J), J=1,NCOL), I=1,NROW)
      IF (NCOL.EQ.16) WRITE (5,160) ((PRTT (I,J), J=1,NCOL), I=1,NROW)
      RETURN
C-----
10   FORMAT (F12.5)
20   FORMAT (2F12.5)
30   FORMAT (3F12.5)
40   FORMAT (4F12.5)
50   FORMAT (5F12.5)
60   FORMAT (6F12.5)
70   FORMAT (6F12.5,/,F12.5,/)
80   FORMAT (6F12.5,/,2F12.5,/)
90   FORMAT (6F12.5,/,3F12.5,/)
100  FORMAT (6F12.5,/,4F12.5,/)
110  FORMAT (6F12.5,/,5F12.5,/)
120  FORMAT (6F12.5,/,6F12.5,/)
130  FORMAT (6F12.5,/,6F12.5,/,F12.5,/)
140  FORMAT (6F12.5,/,6F12.5,/,2F12.5,/)
150  FORMAT (6F12.5,/,6F12.5,/,3F12.5,/)
160  FORMAT (6F12.5,/,6F12.5,/,4F12.5,/)
      END

```



```

C=====
C  SUBROUTINE DATPRT -- DISPLAYS A TWO-DIMENSIONAL ARRAY (16 COLS. MAX)=
C  IN VARIABLE SCREEN PCRMAT FOR USER EASE IN ROW IDENTIFICATION.
C=====

C----- SUBROUTINE MATPRT (PRTT, NROW, NCCL)
C----- IMPLICIT REAL*8 (A-H,O-Z)
C----- DIMENSION PRTT (NROW,NCOL)
C-----

      IF (NCOL.EQ.0)  NCCL=1
      IF (NCOL.EQ.1)  WRITE (5,10)  ((PRTT (I,J),J=1,NCOL),I=1,NROW)
      IF (NCOL.EQ.2)  WRITE (5,20)  ((PRTT (I,J),J=1,NCOL),I=1,NROW)
      IF (NCOL.EQ.3)  WRITE (5,30)  ((PRTT (I,J),J=1,NCOL),I=1,NROW)
      IF (NCOL.EQ.4)  WRITE (5,40)  ((PRTT (I,J),J=1,NCOL),I=1,NROW)
      IF (NCOL.EQ.5)  WRITE (5,50)  ((PRTT (I,J),J=1,NCOL),I=1,NROW)
      IF (NCOL.EQ.6)  WRITE (5,60)  ((PRTT (I,J),J=1,NCOL),I=1,NROW)
      IF (NCOL.EQ.7)  WRITE (5,70)  ((PRTT (I,J),J=1,NCOL),I=1,NROW)
      IF (NCOL.EQ.8)  WRITE (5,80)  ((PRTT (I,J),J=1,NCOL),I=1,NROW)
      IF (NCOL.EQ.9)  WRITE (5,90)  ((PRTT (I,J),J=1,NCOL),I=1,NROW)
      IF (NCOL.EQ.10) WRITE (5,100) ((PRTT (I,J),J=1,NCOL),I=1,NROW)
      IF (NCOL.EQ.11) WRITE (5,110) ((PRTT (I,J),J=1,NCOL),I=1,NROW)
      IF (NCOL.EQ.12) WRITE (5,120) ((PRTT (I,J),J=1,NCOL),I=1,NROW)
      IF (NCOL.EQ.13) WRITE (5,130) ((PRTT (I,J),J=1,NCOL),I=1,NROW)
      IF (NCOL.EQ.14) WRITE (5,140) ((PRTT (I,J),J=1,NCOL),I=1,NROW)
      IF (NCOL.EQ.15) WRITE (5,150) ((PRTT (I,J),J=1,NCOL),I=1,NROW)
      IF (NCOL.EQ.16) WRITE (5,160) ((PRTT (I,J),J=1,NCOL),I=1,NROW)
      RETURN
C----- 10  FORMAT (E12.5)
C----- 20  FORMAT (2F12.5)
C----- 30  PCRMAT (3F12.5)
C----- 40  FORMAT (4F12.5)
C----- 50  FORMAT (5F12.5)
C----- 60  FORMAT (6F12.5)
C----- 70  FORMAT (6F12.5,/,F12.5,/)
C----- 80  FORMAT (6F12.5,/,2F12.5,/)
C----- 90  FORMAT (6F12.5,/,3F12.5,/)
C----- 100 FORMAT (6F12.5,/,4F12.5,/)
C----- 110 FORMAT (6F12.5,/,5F12.5,/)
C----- 120 FORMAT (6F12.5,/,6F12.5,/)
C----- 130 FORMAT (6F12.5,/,6F12.5,/,F12.5,/)
C----- 140 FORMAT (6F12.5,/,6F12.5,/,2F12.5,/)
C----- 150 FORMAT (6F12.5,/,6F12.5,/,3F12.5,/)
C----- 160 FORMAT (6F12.5,/,6F12.5,/,4F12.5,/)
      END

```


LIST OF REFERENCES

1. Hall, W.E., Computational Methods for the Synthesis of Rotary-Wing VTCI Aircraft Control Systems, Ph.D. Dissertation, Stanford Univ., Aug. 1971.
2. Walker, R.A., User's Manual for OPTSYS ⁴ at SCIF, Stanford Univ., Aero/Astro Dept., Dec. 1979.
3. Liu, G., User's Manual for OPTSYS ⁵ at CIT, Stanford Univ., Aero/Astro Dept., Aug. 1982.
4. Bryson, A.E. and Ho, Y.C., Applied Optimal Control, Hemisphere Pub. Co., 1969, (2nd Printing, 1975).
5. Bryson, A.E., "Kalman Filter Divergence and Aircraft Motion Estimators", Jour. Guidance and Control, Vol 1, No. 1, Jan.-Feb., 1978, pp. 71-79.
6. Bryson, A.E. and Hall, W.E., Controller Synthesis for a Rotary-Wing VTCI Aircraft Near Hover, Final Report under NASA Contract NAS2-5143, SUDAAR 419, Stanford Univ., Mar. 1971.
7. Bryson, A.E. and Hall, W.E., Optimal Control and Filter Synthesis by Eigenvector Decomposition, SUDAAR 436, Stanford Univ., Dec. 1971.
8. Wilkinson, J.H. and Reinsch, C., Linear Algebra, Springer-Verlag, 1971.
9. Ogata, K., Modern Control Engineering, Prentice-Hall, 1970.

BIBLIOGRAPHY

Geib, A. and others, Applied Optimal Estimation, M.I.T. Press, 1974.

Kettner, F.I. and Prawel, S.P. Modern Methods of Engineering Computation, McGraw-Hill, 1978.

Kwakernaak, H. and Sivan, R. Linear Optimal Control Systems, Wiley-Interscience, 1972.

Lipschitz, S. and Poe, A., Programming with FORTRAN, Schattau's Outline Series, McGraw-Hill, 1978.

Melsa, J.L. and Jones, S.K., Computer Programs for Computational Assistance in the Study of Linear Control Theory, McGraw-Hill, 1973.

Ogata, K., Modern Control Engineering, Prentice-Hall, 1970.

Research and Educational Association, Problem Solver in Automatic Control Systems/Robotics, 1982.

Sage, A.F., Optimum Control Systems, Prentice-Hall, 1968.

Wilkinson, J.H., The Algebraic Eigenvalue Problem, Clarendon Press, 1965.

INITIAL DISTRIBUTION LIST

	No. Copies
1. Defense Technical Information Center Cameron Station Alexandria Va 22314	2
2. Chairman, Code 67 Department of Aeronautics Naval Postgraduate School Monterey, Ca 93943	1
3. Library, Code 0142 Naval Postgraduate School Monterey Ca 93943	2
4. Professor D.J. Collins, Code 67Co Department of Aeronautics Naval Postgraduate School Monterey, Ca 93943	5
5. CDR V.C. Gordon, Code 67 Department of Aeronautics Naval Postgraduate School Monterey, Ca 93943	1
6. Professor A.E. Bryson Department of Aeronautics and Astronautics Stanford University Stanford, Ca 94305	1
7. LCDR J.G. Holden 12979 Via Del Valedor San Diego, Ca 92129	2

631-807

206451

Thesis

H6432 Hoden

c.l Interactive implemen-
 tation of the Optimal
 Systems Control Program
 (OPTSYSX) on the IBM
 3033.

OCT 18 65
24 FEB 87

31350
30870

206451

Thesis

H6432 Hoden

c.l Interactive implemen-
 tation of the Optimal
 Systems Control Program
 (OPTSYSX) on the IBM
 3033.



thesH6432

Interactive implementation of the Optima



3 2768 002 06814 0

DUDLEY KNOX LIBRARY